

Title	Design and evaluation of an ITS application to control the traffic for emergency vehicles
Sub Title	
Author	Dall'Agnese, Thomas(Dall'Agnese, Thomas) 西村, 秀和(Nishimura, Hidekazu)
Publisher	慶應義塾大学大学院システムデザイン・マネジメント研究科
Publication year	2011
Jtitle	
JaLC DOI	
Abstract	Recently, researchers have been active in the Intelligent Transportation Systems (ITS) development, especially connecting vehicles each other using the Vehicular Ad-hoc Network (VANET). Among the multitude of applications that allow these connected vehicles, a specific attention has been given to traffic flow control and safety purposes. For the emergency vehicles such as ambulances or fire-trucks, to save time by arriving to the destination earlier often means to save lives. For vehicles of such critical purposes, even a gain of a few seconds can be significant. In this research, in the first part we introduce the context of the research, show motivations for the wireless-aided systems, and state the goal of the research. In the second part, we present the current vehicular wireless technology, and introduce ITS applications examples. In the third part, we propose a new ITS application based on a new approach to reduce the time to reach destination of the emergency vehicles, by controlling the traffic, providing information and directions to the drivers surrounding the emergency vehicle, using wireless vehicular communication. We also present a technology alternative to the IEEE 802.11p protocol using the 3G network, quicker to deploy. In the fourth part, we propose an environment to evaluate our application. As the wireless technology is not prevalent yet, we evaluate the benefits of the system depending on the penetration percentage of the wireless technology. We evaluate the system in two scenarios: a highway and a city. In the fifth part, we show that in both cases, the system helps the emergency vehicle to reach its destination faster, especially when more than 60% of the vehicles have the system. We show that thanks to the application, the emergency vehicle can reach its destination at almost the optimal time (when there are no cars in the streets). Finally, we conclude and discuss about the further researches possibilities.
Notes	修士学位論文. 2011年度システムデザイン・マネジメント学 第65号
Genre	Thesis or Dissertation
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO40002001-00002011-0010

慶應義塾大学学術情報リポジトリ(KOARA)に掲載されているコンテンツの著作権は、それぞれの著作者、学会または出版社/発行者に帰属し、その権利は著作権法によって保護されています。引用にあたっては、著作権法を遵守してご利用ください。

The copyrights of content available on the Keio Associated Repository of Academic resources (KOARA) belong to the respective authors, academic societies, or publishers/issuers, and these rights are protected by the Japanese Copyright Act. When quoting the content, please follow the Japanese copyright act.

**Design and Evaluation of an ITS Application
to Control the Traffic for Emergency Vehicles**

Thomas Dall'Agnese

(Student ID Number: 80934612)

Supervisor : Pr. Hidekazu Nishimura

September 2011

**Graduate School of System Design and Management,
Keio University
Major in System Design and Management**

SUMMARY OF MASTER'S DISSERTATION

Student Identification Number	80934612	Name	Thomas DALL'AGNESE
Title Design and Evaluation of an ITS Application to Control the Traffic for Emergency Vehicles			
Abstract <p>Recently, researchers have been active in the Intelligent Transportation Systems (ITS) development, especially connecting vehicles each other using the Vehicular Ad-hoc Network (VANET). Among the multitude of applications that allow these connected vehicles, a specific attention has been given to traffic flow control and safety purposes. For the emergency vehicles such as ambulances or fire-trucks, to save time by arriving to the destination earlier often means to save lives. For vehicles of such critical purposes, even a gain of a few seconds can be significant. In this research, in the first part we introduce the context of the research, show motivations for the wireless-aided systems, and state the goal of the research. In the second part, we present the current vehicular wireless technology, and introduce ITS applications examples. In the third part, we propose a new ITS application based on a new approach to reduce the time to reach destination of the emergency vehicles, by controlling the traffic, providing information and directions to the drivers surrounding the emergency vehicle, using wireless vehicular communication. We also present a technology alternative to the IEEE 802.11p protocol using the 3G network, quicker to deploy. In the fourth part, we propose an environment to evaluate our application. As the wireless technology is not prevalent yet, we evaluate the benefits of the system depending on the penetration percentage of the wireless technology. We evaluate the system in two scenarios: a highway and a city. In the fifth part, we show that in both cases, the system helps the emergency vehicle to reach its destination faster, especially when more than 60% of the vehicles have the system. We show that thanks to the application, the emergency vehicle can reach its destination at almost the optimal time (when there are no cars in the streets). Finally, we conclude and discuss about the further researches possibilities.</p>			
Keywords (5 words): VANET, ITS, V2V, Emergency Vehicle, Traffic Control.			

Table of Contents

I. Introduction	9
I.1 Context.....	9
I.2 Goal of the research.....	15
II. Wireless Vehicular Communications	17
II.1 From Wifi to WAVE.....	17
II.2 Vehicular Ad-hoc Networks.....	20
II.3 Overview of ITS Applications.....	26
III. Definition of the system	38
III.1 Overview of the system.....	38
III.2 Questionnaire and assumptions.....	39
III.3 System specification.....	42
III.4 Responsibility matters.....	55
III.5 Technology alternative.....	56
IV. Evaluation of the system	58
IV.1 Highway Simulation Environment.....	59
IV.2 City Simulation Environment.....	68
V. Results	70
V.1 Highway Simulation.....	70
V.2 City simulation.....	78
VI. Conclusion	86
VII. Further research	87
References	88
Appendices	92

SPECIAL THANKS TO

Nishimura-sensei, for all his support during my 2 years at Keio SDM,

My parents, for always having helped me during my schooling,

My wife, for always being by my side during my hard times in a foreign country,

Mori-san and *Edward*, for being very kind senpai

I. Introduction

I.1 Context

Emergency vehicles such as ambulances, fire-trucks or police cars are special vehicles designated to respond to an emergency. Thus, reaching their destination as fast as possible is the principal concern. For example, an ambulance reaching the place of an accident. An ambulance going back to the hospital carrying an injured person who needs strong medical assistance. A fire-truck which wants to reach a house where a fire started. A police car which wants to reach the place where a crime has just been announced. In all these cases, the time to reach the destination is the most crucial point. Often, reaching the destination a few seconds earlier can make a big difference and even save lives.

In order to reach their destination faster, the emergency vehicles have priority over the normal vehicles on the road and are often permitted by law to break conventional road rules in order to reach their destinations in the fastest possible time, such as (but not limited to) driving through an intersection when the traffic light is red, or exceeding the speed limit, like in the United States of America and in most of the European countries. Breaking the rules and driving aggressively being dangerous, emergency vehicles have sirens, high-visibility markings and warning lights. However, the visibility of the emergency vehicles is very local and the drivers can detect the emergency vehicle only when it gets close to them. As a result, the emergency vehicles are still retarded by the other vehicles. Besides, due to the fact that the emergency vehicle drivers take risk, they are sometimes involved in traffic accidents. According to the U.S. Department of Transportation^[1], 152 people were killed in 2009 in the USA in a crash involving an emergency vehicle (including ambulances, fire-trucks and police cars). This report only includes the fatalities and not the injuries, which are assumed to be much higher. It shows that the emergency vehicles are facing troubles to reach their goal (reaching the destination as fast as possible) and that trying to save lives sometimes engender collateral damages.

The number of fatalities involving emergency vehicles is low compared to the 1.3 million people who die each year on the world's roads and the 20 to 50 million sustain non-fatal injuries^[2]. Road traffic injuries are the leading cause of death among young people, aged between 15 and 44. Actions are needed to make the world's roads safer and all these car accidents lead the car manufacturers to take the problem very seriously.

Nissan Motor Co., Ltd. has been one of the first car makers to concretely take actions through its SKY Project in collaboration with industry majors like NTT DoCoMo, Inc., Matsushita Electrical Industrial Co., Ltd., and Xanavi Informatics Corp. SKY is an abbreviation of "Start ITS from Kanagawa, Yokohama" and the project aims to help reducing traffic congestion, fatalities and injuries^[3].

At first, the SKY project was focusing on safety for children. Children wearing a special equipment can be detected by the system in the Nissan cars and the driver will be vocally and visually warned on the car navigation system if there is a risk of hurting the children. Tests have been conducted between October 2006 and March 2007 and the test vehicles passing through the areas in Kanagawa Prefecture (Japan), where children were equipped, received an audible warning: "Children nearby, please be careful." An icon also appeared on the car's navigation screen carrying the same warning.

The SKY Project has now several applications, such as "Pedestrian Traffic Safety using GPS mobile phone", that uses the pedestrian mobile phones to inform of their presence to the drivers of Nissan cars equipped with the SKY Project on-board system. It can also communicate with infrastructure to warn the driver when entering a school zone for example, as shown in Figure I.1 (right side). Another application shown in Figure I.1 (left side) is the "Intersection Collision Avoidance" that warns a driver at an intersection if there is a vehicle coming at the intersection from another road and which could lead to a collision at the current speed. Other applications of the Nissan SKY Project are "Intelligent Speed Advisory", "Opposite Direction Driving Prevention on Highway", "Dynamic Route Guidance" which eases the traffic congestion utilizing ITS and "Skid Incident Info Service".

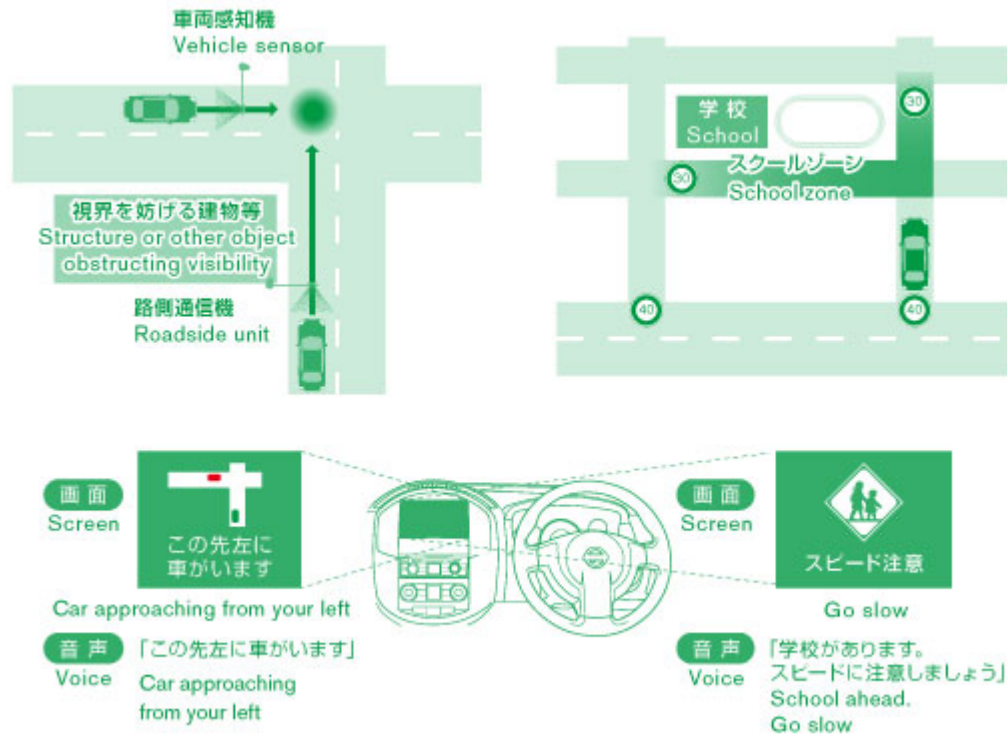


Figure I.1: NISSAN SKY PROJECT Examples

(c) Nissan Motor Co., Ltd.

On one hand, the car manufacturers have been working hard to increase the safety of the roads by developing safety-conscious cars and on the other hand, the technology companies have been working hard to develop a technology that allows reliable communications between the vehicles and the infrastructure and between the vehicles themselves. The network technologies are taking more and more importance in our daily life, day by day. The network of the networks is little by little connecting every single electronic device together, and the vehicles do not escape from this network. While most of the applications offered by the network technologies are for an entertainment purpose, another main goal of the new technologies in general is to improve our daily life, both by entertaining us but also and especially by making our life easier and safer.

In 2004, the Association for Computing Machinery (ACM) organized the first ACM Workshop on Vehicular Ad Hoc Networks (VANET 2004^[4]) that lead to the definition of a

vehicular communication standard by the Institute of Electrical and Electronics Engineers (IEEE), the IEEE 802.11p, also called “WAVE” (Wireless Access in Vehicular Environments). This protocol has been finalized in 2010 so car manufacturers can now implement the technology on their vehicles to allow the vehicles to communicate between them and with the infrastructure through a fast and reliable wireless technology. This technology will be detailed in the next section. WAVE aims to bring “very high speed, rapidly changing, radio environments encountered by cars and trucks”^[5]. Thanks to this technology, vehicles will be able to exchange information between them and with the infrastructure. Figure I.2 represents this kind of network.

Nowadays the information networks (in the computer meaning) are taking a bigger and bigger importance and the vehicles are potential hosts of these networks. Many researchers in universities and industries are working on what is called ITS (Intelligent Transportation System), which refers to the efforts to add information and communications technology to transport infrastructure and vehicles, and to manage factors that typically are at odds with each other, such as vehicles, loads, and routes to improve safety and reduce vehicle wear, transportation times, and fuel consumption. It embraces various technologies and applications.

In Figure I.3, we can see that the number of publications about ITS has been constantly increasing over the years until 2008, when it started to stagnate.

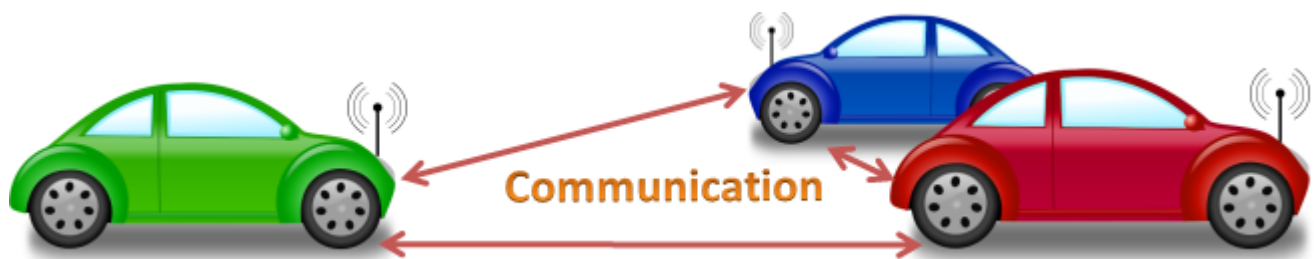


Figure I.2: Wireless communication between vehicles

Number of publications related to ITS over the years

Source: Data collected from Google Scholar

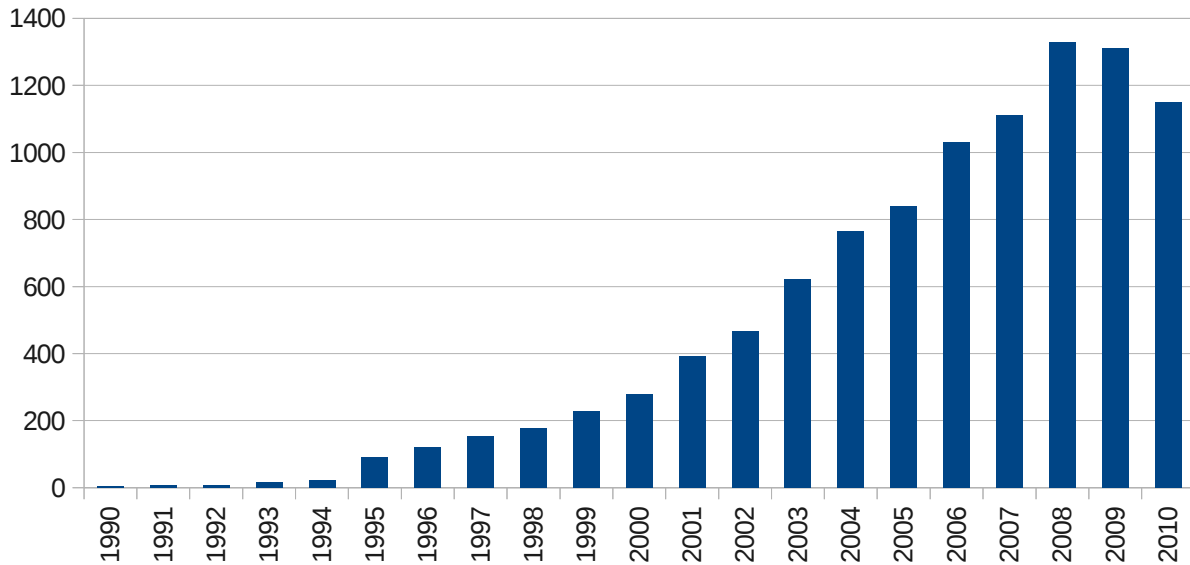


Figure I.3: Researches related to ITS

Number of publications related to V2V over the years

Source: Data collected from Google Scholar

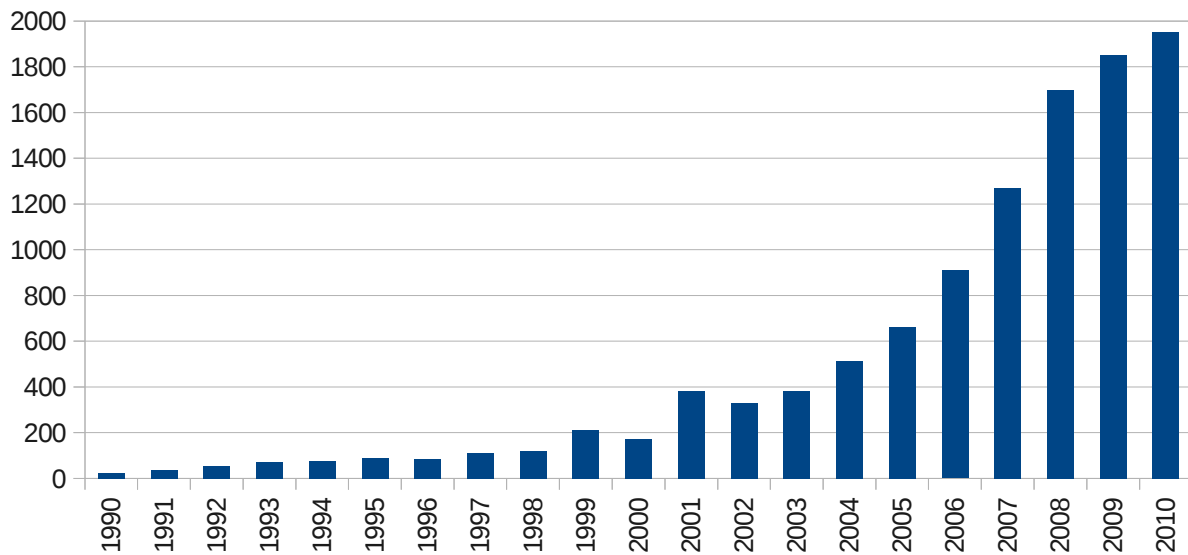


Figure I.4: Researches related to V2V

ITS can rely on different technologies, such as physical sensors, video recognition, computational technologies, cellular communication... But recently, a particular technology is focusing most of the efforts: the wireless communications that are becoming more reliable, faster, wider-ranged, and that can be used to connect the vehicles each others and with the outside (with internet for example). WiMax is a good example of a wireless technology that pops out recently, but the general purpose wireless communications are not the only ones to focus researchers attention^[6]. A wireless communication protocol especially designed for Vehicle-to-vehicle (V2V) and Vehicle-to-infrastructure (V2I) communications has been developed and is now finalized. It gives a good opportunity to car manufacturers to implement the technology in their new cars. This technology will be detailed in the next chapter.

In Figure I.4, we see clearly that the number of publications has been exponentially increasing from and after 2004. Indeed, this date coincide with the first ACM (Association for Computing Machinery) workshop on Vehicular Ad-hoc Networks (VANET 2004) when the base of the communication protocol standard has been discussed.

The researches in V2V communication split into two categories:

- improving the communication protocol (transmission delay, routing...)
- design applications that rely on the vehicular communication (ITS applications)

This study is part of the second category and will introduce a new ITS Application.

Car manufacturers already shown interest and active implementation of the vehicular communication to their recent models.

Cadillac was one of the first company to demonstrate a system using Vehicle-to-vehicle communication in 2006. Other car makers like BMW, Daimler, Honda, Mercedes and Volvo also actively work on vehicular communication and publish regularly papers on the topic.

Nissan also worked very actively on ITS applications with its SKY Projects as explained previously and recently released an electric car called LEAF^[25] that can communicate with Nissan central server through its “Carwings” system. This system is able to transmit the location of the vehicle, its speed and its destination. It is used only to get information about the savings done by using an electric vehicle compared to a traditional car and to read RSS streams, but it shows the interest in the wireless vehicular technologies from the car manufacturers.

1.2 Goal of the research

Assuming that the vehicular communication will be a standard for all the vehicles in the future, many researchers study how to use this information and connectivity to improve the experience and safety of the drivers and passengers. This is referred as “ITS” as introduced previously. Many applications that will be detailed in the next chapter have been studied, such as collision avoidance, traffic jam prevention, adaptable traffic lights...

Most of the ITS applications deal with safety or traffic management. In this study, we will introduce an ITS application that serves both safety and traffic management.

Emergency vehicles are particular vehicles that have strong time constraints. These vehicles try to reach their destination as fast as they can. Often, reaching the destination a few seconds earlier can make a big difference and even save lives.

Emergency vehicles being an important topic, many researches and studies to help them have been conducted. The oldest and most popular one is the traffic light signal control that automatically switched the green light along the path the emergency vehicle is taking. However, using vehicular networks to help the emergency vehicles to reach their destination faster is not a mature topic and

only a few researches have been made so far.

In a research by R. Shirani et al.^[71], some improvements have already been made to allow an emergency vehicle to reach its destination faster using the wireless vehicular networks. However, there are some limits and in the current study, we propose a different approach to improve the time to reach the destination of the emergency vehicles by giving directions to other vehicles to control the traffic, in order to facilitate the way to the emergency vehicle. Usually, when drivers notice an emergency vehicle, this one is already close and it is often hard, or too late, to make an action to help the emergency vehicle. Providing information about the emergency vehicles in advance, through the vehicular wireless communication, and giving directions to the drivers can help the drivers to make the right action to facilitate the way to the emergency vehicle. As a result, the emergency vehicle would be able to reach its destination faster.

At the same time, we take into consideration the fact that not all the vehicles are equipped with the VANET technology and analyze the impact on the system.

The problem in all other researches is that they assume that all the vehicles have the vehicular wireless communication equipment. However, this will not be true until many years. Thus, we believe it is important to analyze the impact of the system depending on the percentage of the vehicles equipped with it, assuming that in a near future, only a small percentage of cars will be equipped with VANET.

Therefore, we will evaluate the system with variable percentage of vehicles equipped with it.

In this dissertation, we will first introduce the vehicular wireless communication and some of its ITS applications. Then, we will define our system, evaluate it and discuss the results. Finally we will conclude and give an overview of the future researches possibilities.

II. Wireless Vehicular Communications

To understand how the system works, a clarification of the technology it relies on is called for.

II.1 From Wifi to WAVE

In the past decades, the wireless network technologies kept improving and are now part of our daily life. All the current laptops have a Wifi interface (a, b, g or n) and there are more and more Wifi access points, especially in the cities. For example there are now Wifi access points in some restaurant chains, in most of the cafes, in some malls, in some electronic shops, at the airports, at schools, at some train or subway stations... Recently the Internet Service Providers (ISP) are deploying the WiMax technology, a recent evolution of the Wifi that has a wider communication range and whose goal is to cover whole cities and countries.

An access point (AP) is a relay to another network. The most popular usage of Wifi communication is to connect to an access point which is most of the time connected to Internet. Figure II.1 is a schematic representation of users connecting to a wireless network. The access point is a relay to the internet for the computers.

The blue arrows represent a communication between the distant computers and the access point. The principal advantage of such communication is that when the access point has access to internet, it can share it to the other computers that connect to the access point so the other computers can have access to internet too. This is the most popular usage of the access points, that most of the people do without knowing the details. However, this is not the only usage of the access points and there is another usage that most of the lambda users do not need to know or just do not know, which is to establish a network so that the computers connected to this access point can communication each others (instead of communicating individually with the internet). In this case, the fact that the access points has or share internet is not relevant as the goal is to reach other computers of the network, for example to exchange data (files), resources (CPU, memory) or devices (printers, hard disks). This kind of communication can be represented as shown in Figure II.2.

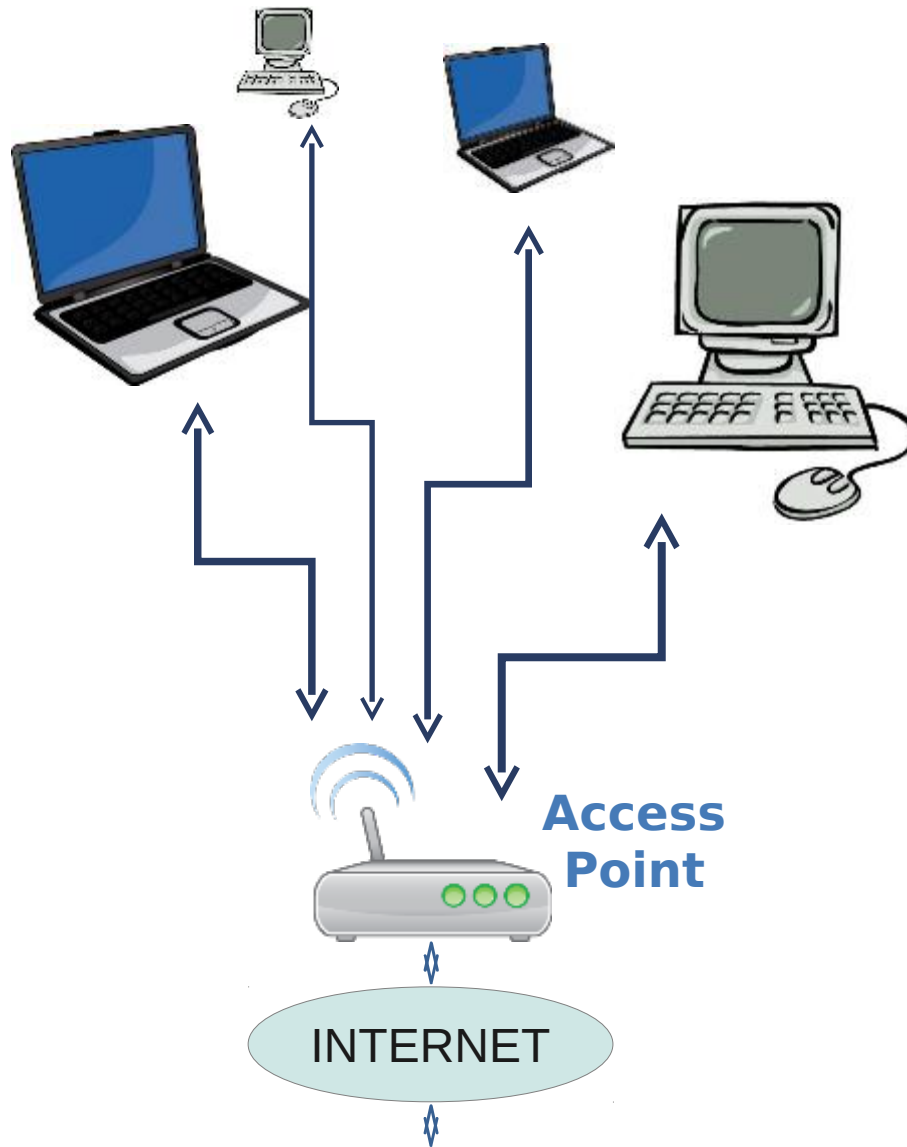


Figure II.1: Representation of a wireless access point

The red arrows in Figure II.2 represent a connection from a computer to another one, through an access point. In this case, the goal is not to get internet from the access point but to exchange data with another computer of the network. A common usage is to exchange files between the computers.

Finally, there is a third usage of the wireless networks, that does not make use of an access point. Indeed, as the computers have a wireless interface, they can establish a direct connection between them through this connection, without the need of a dedicated access point. A representation of such connection is shown in Figure II.3.

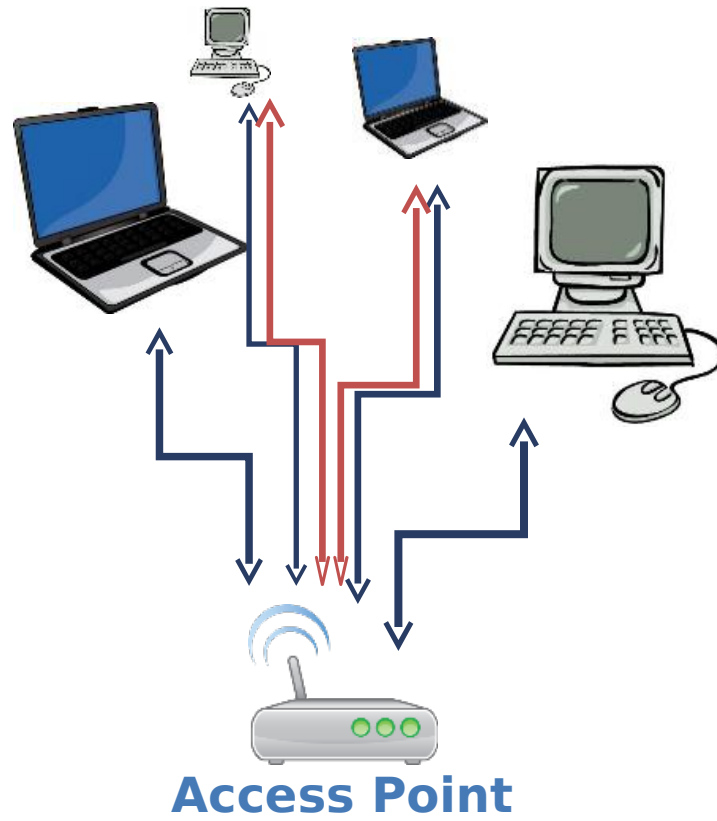


Figure II.2: Computer to computer communication through an access point

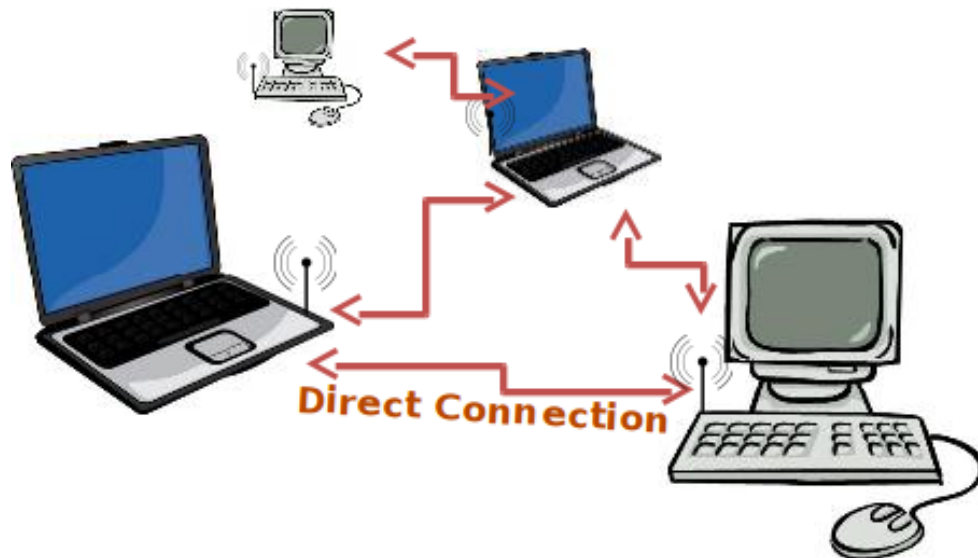


Figure II.3: Direct connection between computers using Wifi

This third kind of network is called a wireless ad-hoc network. This is a decentralized type of wireless network. It does not rely on a preexisting infrastructure like routers or access points. Instead, each node (a node is general term defining a member of a network, that can be a computer or any device with a wired or wireless interface, such as printers, IP cameras...) participates in routing by forwarding data to other nodes.

The obvious main advantage of wireless networks over wired networks is the mobility it provides, as the connected nodes do not need a cable to communicate each others. Besides, they can easily move while there are connected, and connect to different networks without having to change any cable.

This mobility attracted new usage of the wireless networks. As any electronic device with a wireless interface can be part of a wireless networks, new devices started to join the networks, like the vehicles.

Imagine that instead of being computers connected each others, it was your car connected to other cars (or with other devices that have access on the network) through a wireless interface. Figure I.2 (Wireless communication between vehicles, page 12) represents this kind of network.

II.2 Vehicular Ad-hoc Networks

A Vehicular Ad-Hoc Network, or VANET, is a form of Mobile ad-hoc network (MANET), to provide communications among nearby vehicles and between vehicles and nearby fixed equipment, usually described as road-side equipment.

We can differentiate two types of communications:

- V2V for Vehicle-to-Vehicle (or sometimes referred as “C2C” for “car-to-car”)
- V2I for Vehicle-to-Infrastructure (or sometimes referred as “C2I” for “car-to-infrastructure”)

Figure II.4 illustrates the two different types of communications.

V2V is a direct communication a vehicle and another vehicle on the road and is similar to the case previously shown in Figure II.3 (Direct connection between computers using Wifi) except the computers are not computers but vehicles in the V2V case.

V2I is a communication between the vehicles and devices on the road referred as “road-side units” (RSU) that can communicate with the vehicles (to transmit data, information, warnings, provide internet... to the vehicles) and is similar to the case previously shown in Figure II.2(Computer to computer communication through an access point) except the computers are not computers but vehicles in the V2I case, and the access point is a road-side unit.

The main goal of VANET is to provide safety and comfort to passengers. To this end, a special electronic device is placed inside each vehicle that provides ad-hoc network connectivity for the vehicles and its passengers. This network tends to operate without any infrastructure or legacy client and server communication. Each vehicle equipped with VANET device is a node in the ad-hoc network and can receive and relay others messages through the wireless network. Collision warning, road sign alarms and in-place traffic view will give the driver essential tools to decide the best path along the way.

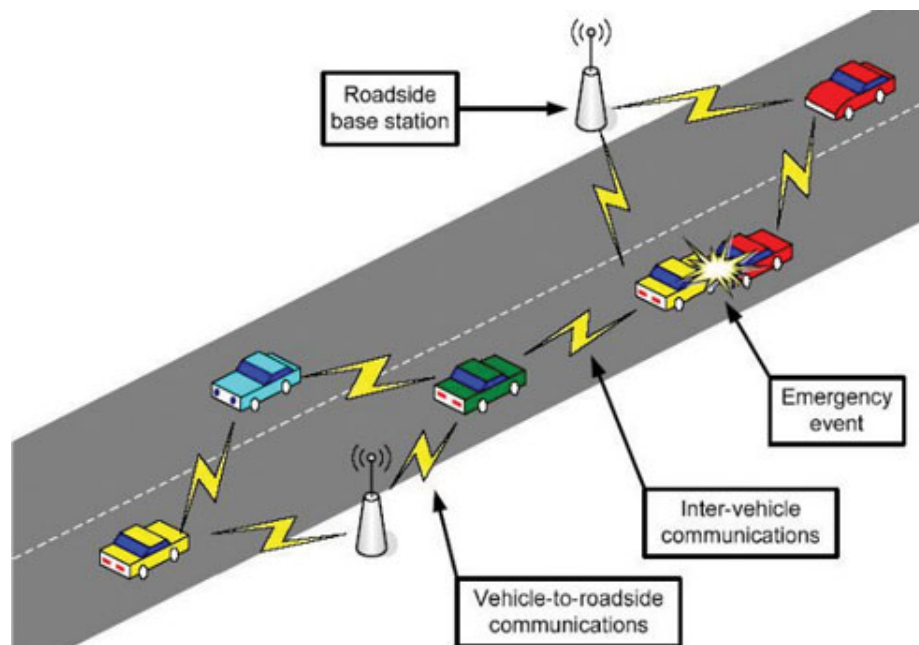


Figure II.4: Schematic Representation of Vehicular Ad-hoc Networks

© Mainak Ghosh & Sumit Goswami, IIT, Kharagpur

There are also multimedia and internet connectivity facilities for passengers, all provided within the wireless coverage of each car. Automatic payment for parking lots and toll collection are other examples of possibilities inside VANET.

Most of the concerns of interest to MANET (Mobile Ad-hoc Networks) are of interest in VANET, but the details differ^[23]. Rather than moving at random, vehicles tend to move in an organized fashion. The interactions with roadside equipment can likewise be characterized fairly accurately. And finally, most vehicles are restricted in their range of motion, for example by being constrained to follow a paved highway.

In MANET in general, the nodes are not supposed to move a lot. For example, when someone using a laptop connects to a Wifi access point, he or she usually does not move once connected.

In VANET, the nodes are vehicles, which are almost always moving.

Vehicular Ad-hoc Networks are expected to implement variety of wireless technologies such as Dedicated Short Range Communications (DSRC) for short range and IEEE 802.11p for wider range. Other candidate wireless technologies are Cellular, Satellite, and WiMAX. Vehicular Ad-hoc Networks can be viewed as component of the Intelligent Transportation Systems (ITS).

Currently, DSRC technology is used in Europe and Japan for electronic toll collection (ETC) and aims to eliminate the delay on toll roads by collecting tolls electronically.

The recent NISSAN LEAF 100% electric car has a so-called "Leaf Carwings" system that connects to the internet through the GSM cellular connection of the driver's mobile phone. It is currently used to feed energy economy statistics to a central server and also allows the drivers and passengers to read RSS feeds (such as CNN, Fox News, a weather channel...).

Vehicular Ad-hoc Networks are a cornerstone of the envisioned Intelligent Transportation Systems (ITS). By enabling vehicles to communicate each others via Inter-Vehicle Communication (IVC) as well as with roadside base stations (or road-side units) via Roadside-to-Vehicle Communication (RVC), vehicular networks will contribute to safer and more efficient roads by providing timely information to drivers and concerned authorities. The interesting research area of Vehicular Networks is where ad-hoc networks can be brought to their full potential.

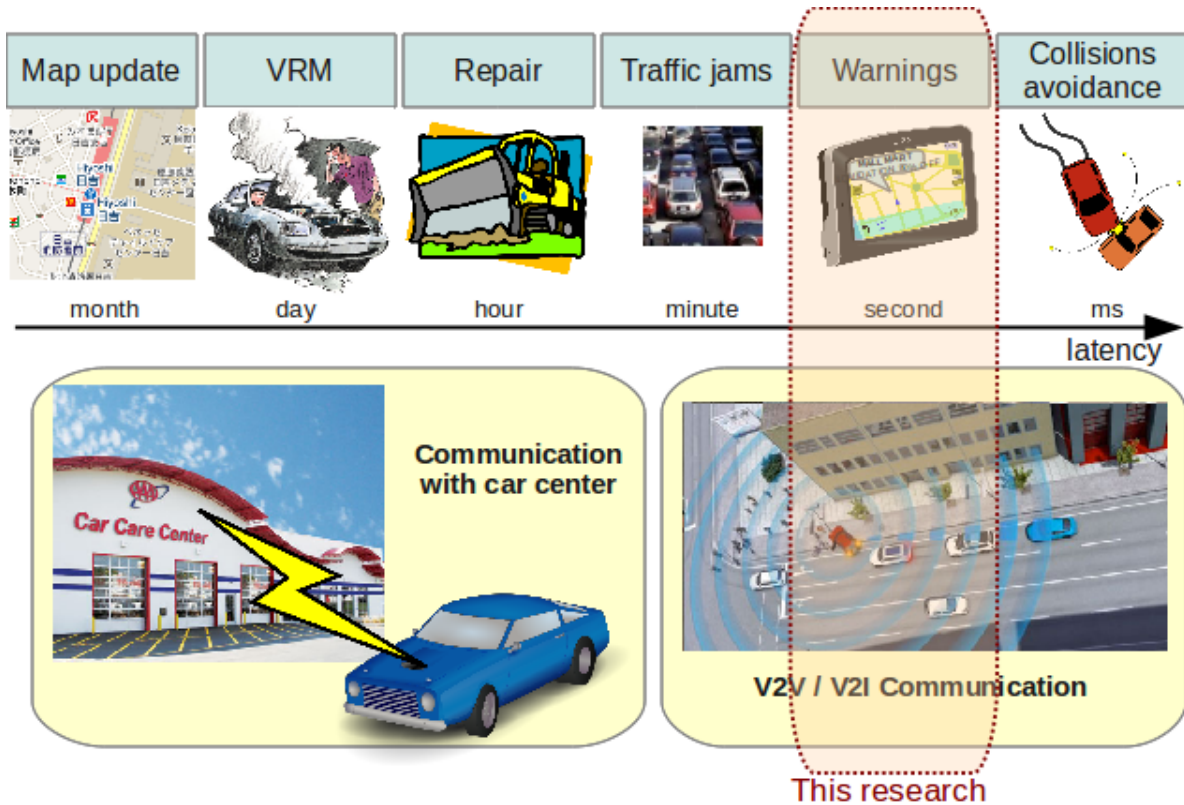


Figure II.5: ITS Applications Classification

An ITS Communication System refers to the efforts to add information and communications technology to transport infrastructures and vehicles. Figure II.5 is an overview of the ITS applications.

During this study, we will present an ITS application that gives guidance and warnings to the drivers. Thus, this study belongs to the “warnings” category and require a small delay of transmission, but not as small as for collisions avoidance-like systems.

In order to achieve this goal, the V2V (vehicle-to-vehicle) communications are done through the WAVE protocol (Wireless Access in Vehicular Environments) which has a lower latency than the Wifi or the WiMAX protocols. Figure II.6 compares the different popular wireless protocols (WAVE, Wifi, GSM and WiMax) in terms of communication range and delay (time to connect plus latency)^[8].

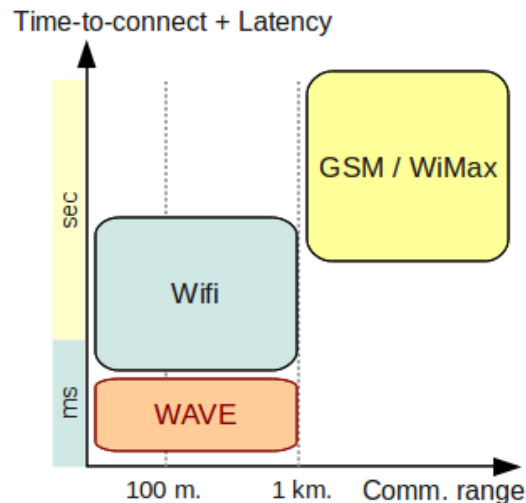


Figure II.6: Delays of the main wireless protocols

“Wifi” refers to the IEEE 802.11 'a', 'b', 'g' and 'n' protocols, that are widely used by laptops, cellphones and tablets. The 'a' and 'b' protocols are the oldest ones (1999) and the first ones to be widely deployed in the laptops. In 2003, the 'g' protocol has been finalized and allowed higher data transmission rate (up to 54 Mbps, to which earned it the popular name “Wifi 54”). However, the coverage range was similar to the protocols 'a' and 'b' (about a hundred meters indoor, 400 meters outdoor). More recently, in 2009, the 'n' protocol was released and allowed a higher data transmission rate (up to 150 Mbps). In addition to a higher data transmission rate, the 'n' protocol doubled the coverage range (about 200 meters indoor and 800 meters outdoor).

WAVE refers to the particular IEEE 802.11 'p' protocol that was designed especially for vehicular communications. It focuses on a small time-to-connect and latency while keeping a communication range up to 1 kilometer^{[13][14]}.

The IEEE 802.11p protocol has been finalized in July 2010 and aims to bring wireless in “very high speed, rapidly changing, radio environments encountered by cars and trucks”. The delay (time-to-connect + latency) is at maximum 100 ms^[18].

WAVE is only a part of the VANET protocol stack (the physical layer (PHY) and MAC layer^{[20][21]}) as shown in Figure II.7.

Application (Resource Manager)	IEEE 1609.1	
Application (Security Services)	IEEE 1609.2	
UDP/TCP	WSMP	IEEE 1609.3
IPv6		
LLC	IEEE 802.2	
WAVE MAC	IEEE 1609.4	
WAVE PHY	IEEE 802.11p	

Figure II.7: VANET Protocol Stack

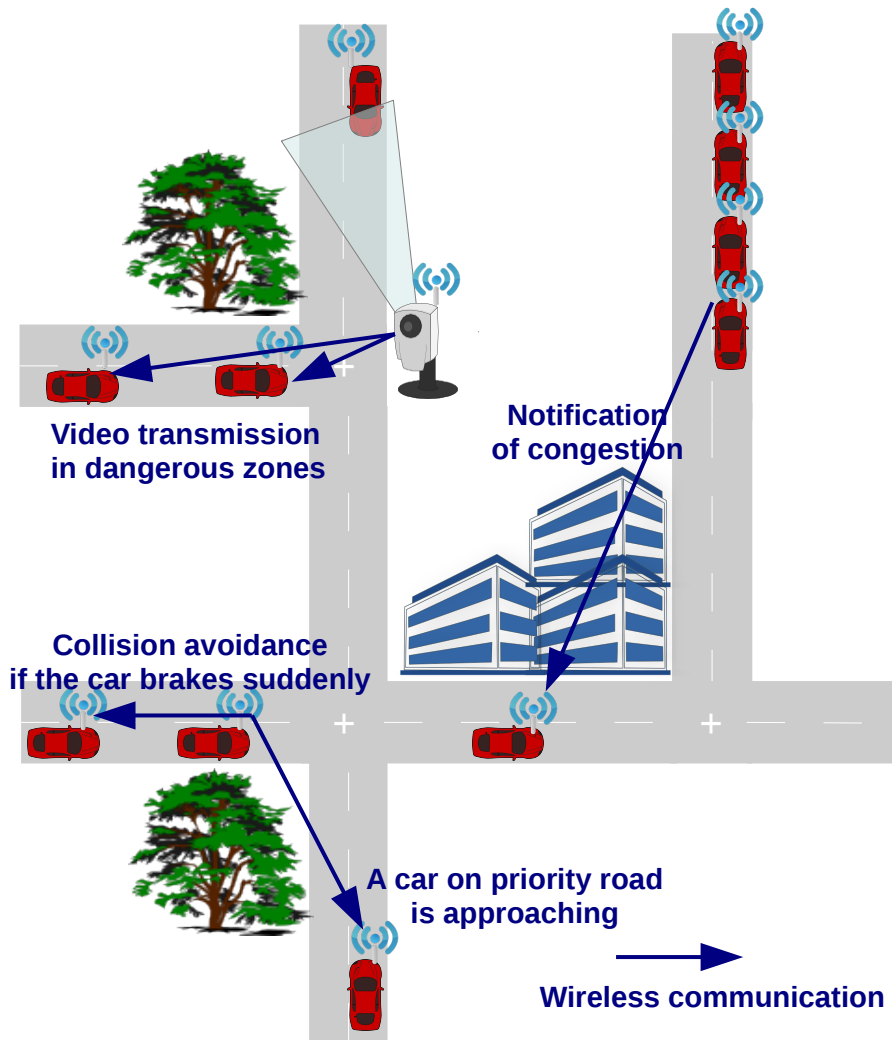


Figure II.8: Example of some ITS Applications

Finally, WiMAX refers to the IEEE 802.16 'm' protocol. It provides a data rate of 100 Mbps and plans to reach 1 Gbps^[15]. This protocol can cover many kilometers, but its time-to-connect and latency are too high for being reliable for vehicular communication at the moment. Same problem with GSM cellular communication whose latency is too high.

An example of what can be achieved thanks to the Vehicular Ad-hoc Network is illustrated in Figure II.8. This kind of communication is a localized communication, and it is blindingly obvious that the destination of the message has to be identified by its location.

In the next section, we will see some ITS applications in more details.

II.3 Overview of ITS Applications

In the last decades, the number of vehicles on the roads has been quickly increasing, increasing by the same time the congestion of the roads. Considering the fact that the vehicles can communicate each others and with the exterior opened a multitude of new possibilities.

Among the possibilities, current researches have been focusing on two purposes: safety^{[22][24]} and traffic flow control^[9]. We will now see some of them, relying on wireless technology or not.

a) For safety purpose

Sudden brake warning

A sudden brake warning system^[29] can inform following vehicles of a sudden brake/halt. If a vehicle suddenly halts, with a strong brake for example if an animal suddenly crosses a street, the vehicles following this vehicle have a non negligible chance of colliding with this vehicle, especially if there are close to the vehicle, driving at high speed.

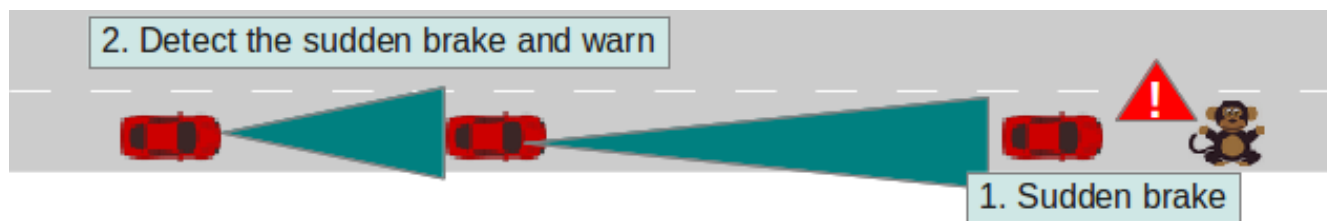


Figure II.9: Sudden brake warning with sensors



Figure II.10: Sudden brake warning with wireless communication

Even if a physical sensor on the following vehicles that detects the preceding vehicle and calculate a time to collision might be more reactive, such ITS application can also be based only on vehicular wireless networks, and in that case does not need high-technology physical sensors, has a wider communication range than the sensors range and is not affected by the weather conditions.

In case of a strong brake, a vehicle with a “sudden halt warning system” sends a broadcast warning message to warn the following vehicles. The following vehicles then receive a warning signal and can react faster to such situation.

Report accidents

The vehicular wireless communication can be used to report accidents to the surrounding vehicles. It can avoid series of accidents by warning the vehicles that are reaching the accident location and can also be used to redirect the oncoming vehicles if the road has been blocked because of the accident, to limit traffic jams in this junction.

The vehicles that received the warning can change their route to avoid the accident location^[30].

This example is shown in Figure II.11.

Warnings at intersections, entering highways and lane changes

When arriving at intersections with low visibility, your vehicle can warn you if there is an approaching vehicle^[31] so you can react and avoid a collision as shown in Figure II.12.

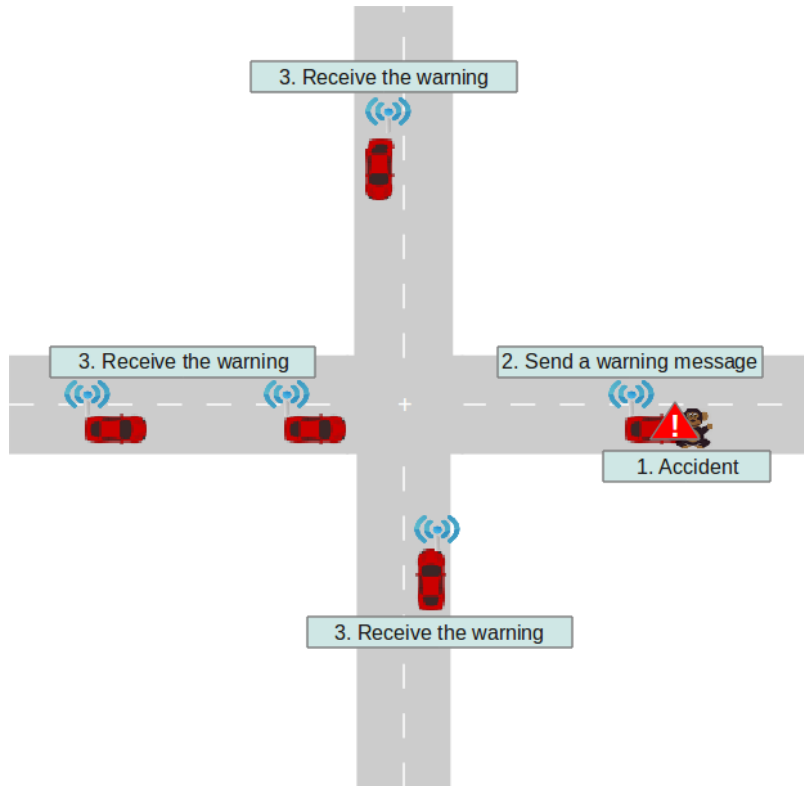


Figure II.11: Accident report example

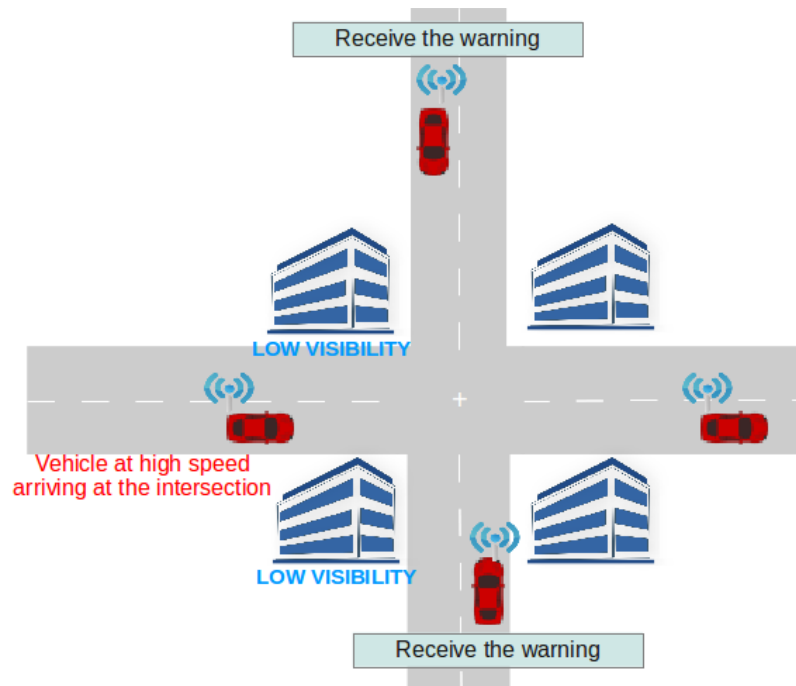


Figure II.12: Example of a warning at an intersection

The same warning can occur when arriving in a highway if there is already another vehicle arriving.

Also, when you try to change lane, the vehicle can warn you if there is an approaching vehicle that you did not notice. This can also be done with physical sensors, which might be more efficient, but using vehicular wireless networks has the advantage of not needing high-technology sensors and not being affected by the weather conditions.

Obstacles discovery

Obstacles discovery is similar to the sudden brake warning system except it warns you when an obstacle is on the road. The difference with the “sudden brake warning system” is that the obstacle is not a vehicle anymore, and probably not having a wireless interface that can warn the other vehicles. So there are two ways of discovering obstacles: with physical sensors, or through warnings from other vehicles or road side units that detected the obstacles.

While the sensors are individual and efficient for big obstacles (like a rock on the street or something that fall out from the preceding vehicle), they cannot detect obstacles such as water on the street, dirt or a broken road.

Road-side units, which are devices dedicated to communicate with the vehicles, can be deployed where there is an unusual obstacle and warn the oncoming vehicles. A driver who detected an obstacle can also manually decide to broadcast a warning message that will also be broadcast by the other vehicles around the obstacle location.

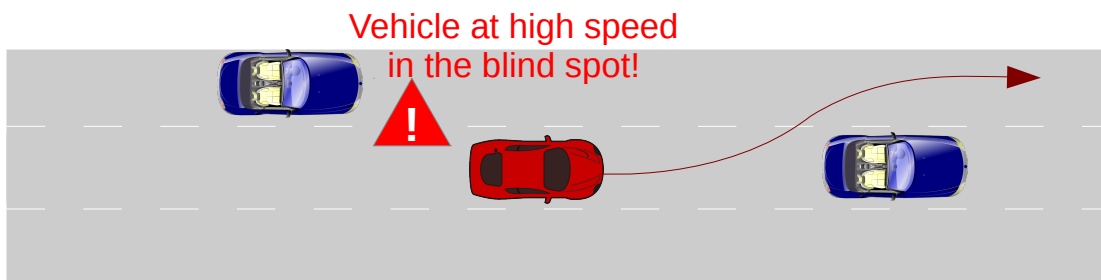


Figure II.13: Blind spot detection / Lane-change assistant



Figure II.14: Locality information example

b) Travel-related and locality information

Road-side units can provide more detailed and updated information about the location the road-side unit is deployed to. A road-side unit can provide a detailed map of the locality to the vehicle that displays it to the driver on the car navigation system.

It can add information about the locality to the current car navigation system information, like business locations, restaurants menu and prices, shops discounts, etc...

It can inform the driver of the closest car services or gas stations, and its prices.

An example of a message displayed on the car navigation system announcing a liquidation in a local shop is shown in Figure II.14.

c) General services

Internet access

General services can also be brought to the vehicle, like internet in general, if the road-side units are connected to the Internet and share it. This situation becomes similar to the

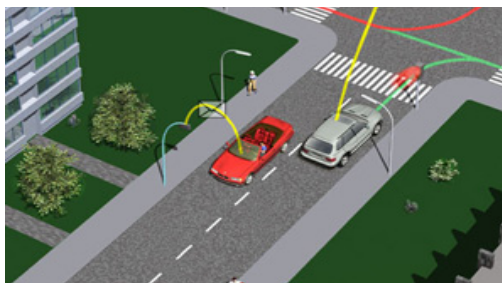


Figure II.15: Example of e-mail checking with V2I

Figure II.1 (Representation of a wireless access point, page 18) except that the computers are not computers in this case but vehicles, and the access point is a road-side unit. By accessing Internet through the road-side units, passengers can for example surf on the web or check their emails (as shown in Figure II.15).

Gaming, chat, sharing

When there are no road-side units, two or more vehicles can still communicate each others, allowing to share data in the same way as in the Figure II.3 (Direct connection between computers using Wifi) except the computers are not computers in this case but vehicles.

This can allow for example two passengers in two different vehicles to play a video game together, chat together or share files.

d) Traffic Management

Variable speed limits

Variable speed limit is a traffic management application to limit the traffic jams in highly congested streets. Instead of being a fixed traffic sign, the speed limit traffic signs are electronic traffic signs that calculate the speed limit depending on the road congestion. For example if a road is highly congested, the speed limit will be reduced ahead in the street to avoid a traffic jam^[33]. This kind of system can also limit the accordion effect (series of low congestion and high congestion) in highways by setting a lower speed limit if the risk of accordion effect is high.

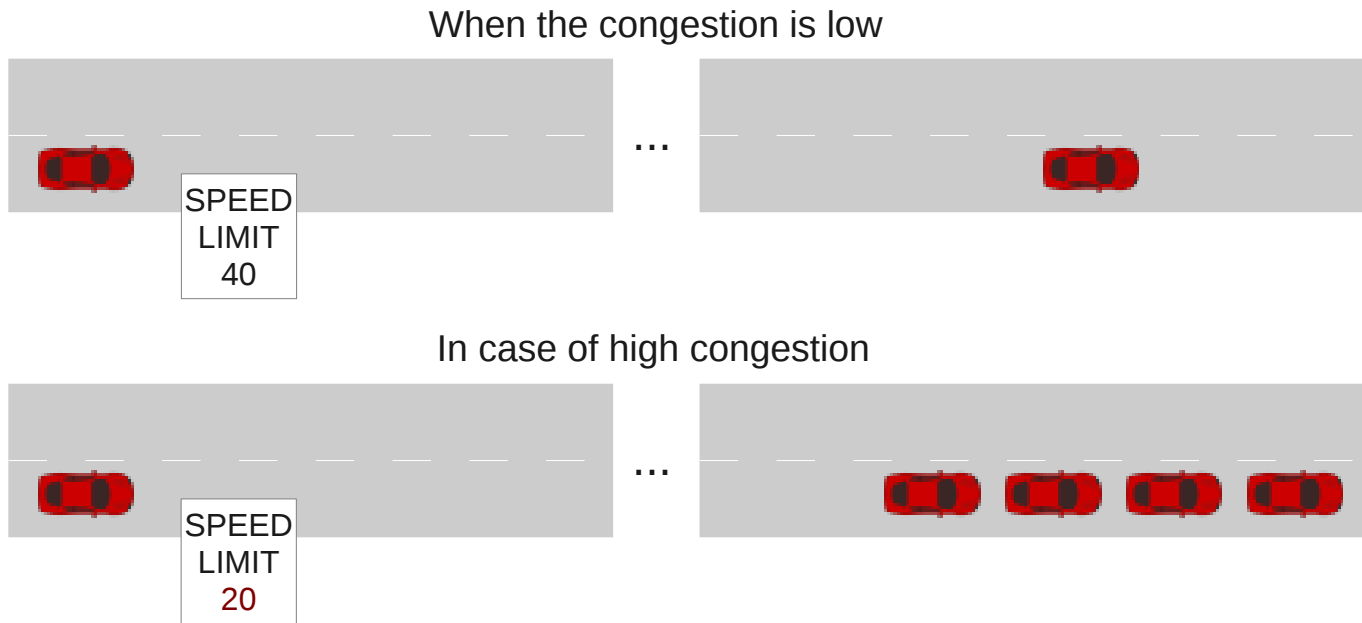


Figure II.16: Simplified schema of the variable speed limit system

To be aware of the road congestion, physical sensors can be installed on the road (as this is the case now in some highways in the USA) or a wireless interface can be added to the variable speed limit traffic sign to communicate with the vehicles on the road and get the road congestion information (as it is expected to be in the future).

Adaptable traffic lights

Adaptable traffic light systems^[11] are close to variable speed limit systems. Instead of varying the speed limit, the duration of the traffic lights are dynamically assigned. For example if at an intersection of two roads, one is highly congested whereas the other one is not, the green light will be longer on the congested one than in the other one.

This kind of system can be done with sensors on the street to detect the vehicles, or using vehicular wireless communication. The advantage of using vehicular communication is, in addition to the fact that you do not need physical sensors, that you can collect data from larger distances and from nearby intersections to adapt more efficiently nearby traffic lights together.

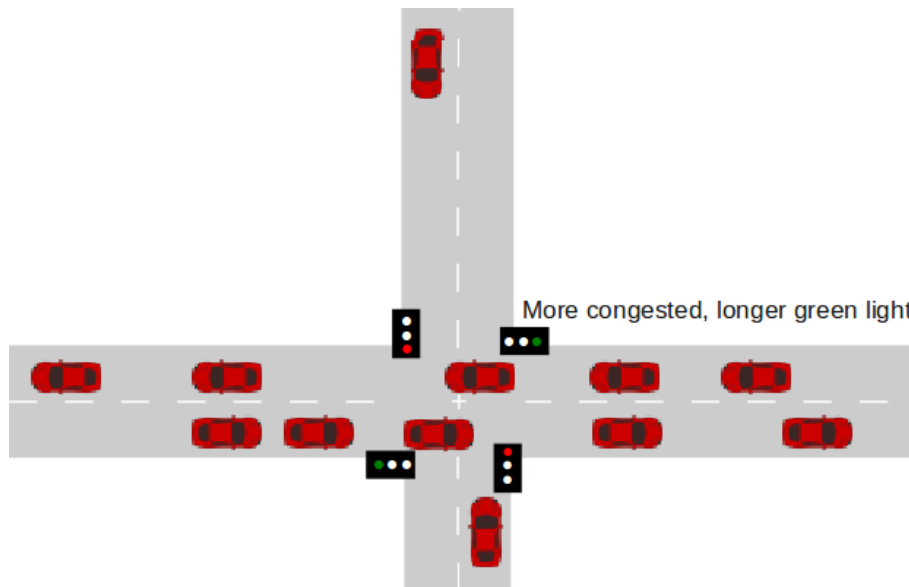


Figure II.17: Representation of a simplified adaptable traffic light

Figure II.17 is a represents a simplified adaptable traffic light system to illustrates it.

The adaptation is dynamic, so if later, the other street becomes more congested, the other street will have a longer green light. Adaptable traffic lights do not rely only on the street congestion like in the above drawing and complex systems have been studied, especially for intersections with multiple lanes for different directions, with different traffic lights per lane.

Directions and route optimizations

Directions and route optimizations is a major field of improvement of the vehicular wireless communications. Thanks to the vehicular communication, the location, speed and direction of the vehicles in an area can be known. This information can be used to calculate the shortest path to a given destination, taking into consideration the congestion of the roads^[33].

There are two ways of doing so: a wireless-equipped vehicle calculates autonomously a path to its destination knowing the surrounding vehicles (efficient for short destinations only) or via a centralized road-side unit, if present, that has more information in a wider area about the vehicles, that calculates the shortest path and returns it to the vehicle.

Figure II.18 shows a simplified route determination scenario.

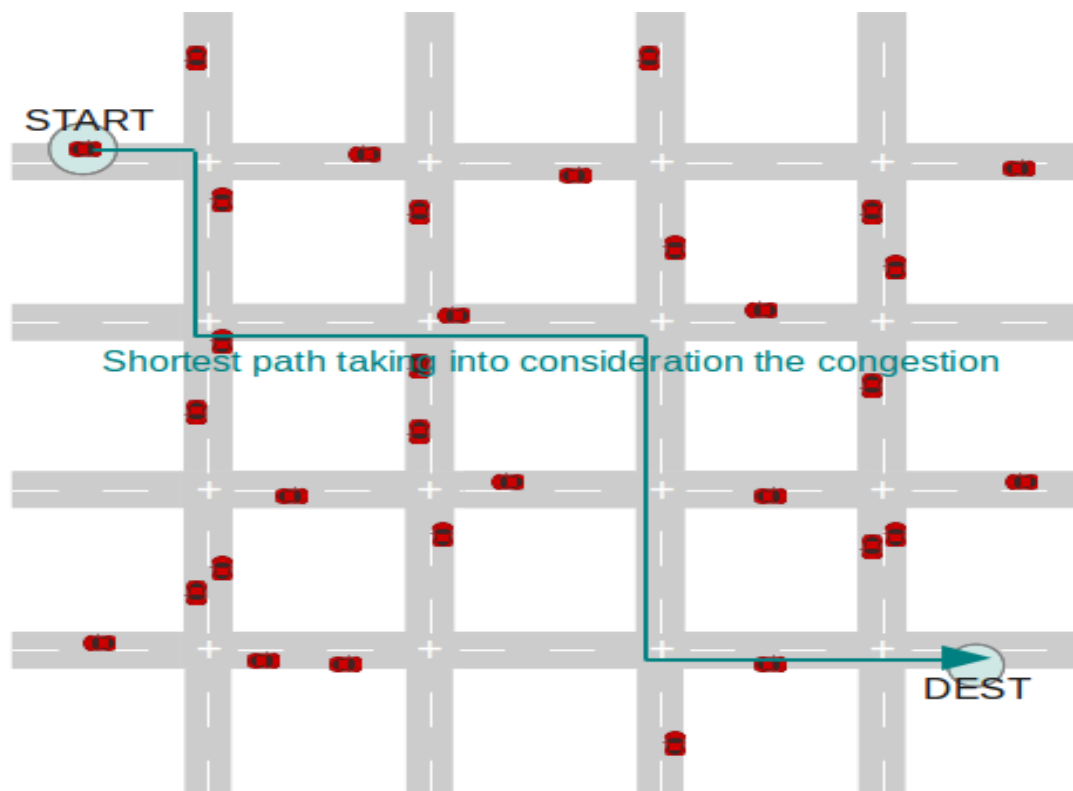


Figure II.18: Representation of a simplified shortest path calculation

Shortest path calculation taking into consideration the changes of the congestion over the time and anticipating future states is not an easy task and focuses the attention of researchers in various fields.

Obstacles warning

We previously introduced obstacles discovery for safety purposes, but obstacles discovery can also be used for traffic management. In this case, the obstacle is not detected by physical sensors but announced by road-side units or by vehicles themselves. For example if a furniture van is blocking a street, or if there was an accident previously so now a street is blocked, it can broadcast this information to surrounding vehicles so they do not engage in this street but choose another way to reach their destination. Figure II.19 represents this kind of scenario.



Figure II.19: Obstacle warning example

Adaptive Cruise Control

Adaptive Cruise Control (ACC)^{[16][17]} is the action of autonomously adapting the speed of the vehicle based on the preceding vehicles. We can differentiate two types of ACC, which correspond to two ways of collecting information from the preceding vehicle(s): based on sensors (through physical sensors) or based on network data (through vehicular wireless communication, called W-ACC for Wireless-ACC)^[10]. Both types of ACC have advantages and drawbacks. While the sensor-based ACC is more accurate than the network-based ACC, it can only detect the vehicle ahead while the W-ACC can get information from multiple vehicles from a larger distance.

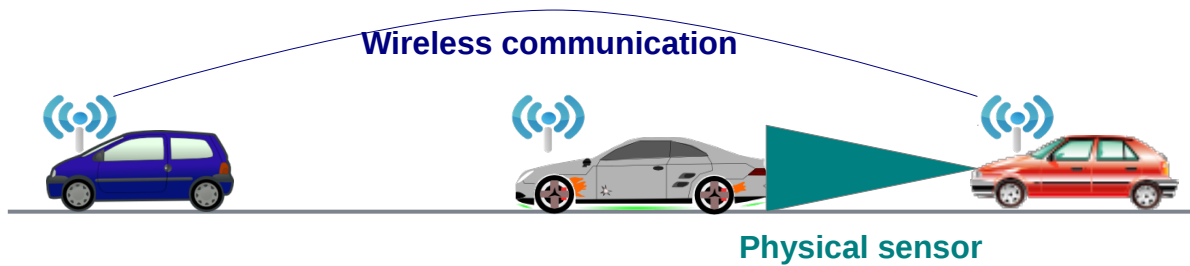


Figure II.20: Schematic Representation of Adaptive Cruise Control

A combination of both technologies allows a better adaptation of the speed of the vehicle. Figure II.20 illustrates these technologies.

Accommodating emergency vehicles

Accommodating emergency vehicles is a specific application of directions and route optimization that focuses on facilitating the way to the emergency vehicles.

This is a traffic management system, but for safety purpose.

Emergency vehicles are special vehicles that need to reach their destination as fast as possible. For example, an ambulance that has to reach the place of an accident or bring an injured person back to the hospital. A fire-truck that has to go to a fire location to stop a fire. A police car that wants to reach a house where a crime has just happened. For all these vehicles, it is important that they reach their destination as fast as possible, and, sometimes, reaching the destination even a few seconds only earlier can make a big difference. Thus, studies to facilitate the way to the emergency vehicles have been done. For example, traffic light adaptation, to give the priority to the roads the emergency vehicle is taking. Such system turns automatically the traffic lights the emergency vehicle is passing through to the green light so the emergency vehicles can avoid stopping at a red light.

Another research by R. Shirani et al. entitled “Absolute Priority for a Vehicle in VANET”^[7] aims at reducing the imposed traveling time of an emergency vehicle using vehicular

communication. In that research, they consider an ambulance which wants to go back to the hospital. All the cars being assumed to have the wireless communication technology periodically broadcast their position, direction and velocity. Based on this data, especially the velocity of the vehicles in each street, the ambulance calculates the shortest path to the destination with a Dijkstra algorithm.

They showed that thanks to that system, the ambulance could reach the destination faster than without the system, especially with highly condensed streets. For example with an average number of vehicles of 50 per kilometer, the time to reach destination was reduced to three quarters of the time to reach destination without the system.

We will define our system in the next chapter.

III. Definition of the system

III.1 Overview of the system

The research by R. Shirani et al. has some limits. It uses the VANET communication only to read the location and velocity of the other vehicles and calculate independently the shortest path to a destination. It does not interact with the other vehicles.

However, we think that considering we have the possibility to communicate between vehicles, we should use this communication to give information to the other vehicles about the emergency vehicle, to take advantage of this “connectivity” between vehicles more efficiently.

We propose in this study a system which is more “collective” (by opposition to the current “individual” approach), allowing the surrounding vehicles to actively participate in facilitating the way to the emergency vehicles.

We believe (which is confirmed by a questionnaire that will be introduced in the next part), that if the drivers surrounding the emergency vehicle knew the location of the emergency vehicle, even before they actually see it or hear its siren, and knew the route the emergency vehicle was going to take, it could help them facilitating the way to the emergency vehicle, thus allowing the emergency vehicle to reach its destination faster.

Such feature is now possible thanks to the VANET communication^[19] combined with the car navigation system. Imagine that in case of an emergency, the emergency vehicle broadcasts a packet (sends a message) repeatedly, letting other surrounding cars which also have VANET equipment know the current location and planned route of the emergency vehicle.

In addition to informing the surrounding drivers of the current location and planned route of an emergency vehicle, we also add some data processing to first check that we are concerned by this emergency vehicle (not to annoy the drivers with useless information), and also to determine if there is an action the driver could do in order to help the emergency vehicle (stop at the next intersection, change lane, change the planned route...). If such action is found, we suggest it to the driver.

III.2 Questionnaire and assumptions

In order to better target the current situation for the emergency vehicle drivers and for the normal drivers towards the emergency vehicles, we went to Kanagawa Prefecture Ward Office (where the firemen and ambulance drivers are) to interview real emergency vehicle drivers and get feedback from them on the system and conduct a survey of 25 questions: 19 questions for normal vehicle drivers and 6 questions specific to the emergency vehicle drivers.

The full questionnaire can be found in Appendix A. 114 people answered to the survey entirely (including 51 emergency vehicle drivers) and the results are summarized as follows.

a) Normal vehicle drivers

First, drivers are often confronted with emergency vehicles. More than 70% of the respondents are confronted with an emergency vehicle once or more than once a week. Among these drivers, they all are willing to help the emergency vehicles. 90% of the respondents always try to facilitate the way to the emergency vehicles in highways while the other 10% often try to. In cities, the results are even more significant with 95% of the respondents always trying to facilitate the way to the emergency vehicles (and 5% often). So during this study, we consider that people are willing to help the emergency vehicles and assume that if they receive notifications about how to help the emergency vehicle, they will follow the indications or at least take it into consideration.

Then, we highlighted that 57% of the respondents need more than 3 seconds to locate an emergency vehicle on a highway. Only 9% of them can locate an emergency vehicle after hearing the siren within 2 seconds. In cities, it seems easier to locate the emergency vehicles, where almost 50% of them can locate it within 3 seconds. 19% of them still need more than 5 seconds. This can be explained by the fact that in city, when one hears a siren close to an intersection, it is hard to know which street is the emergency vehicle coming from and going to. So even if one hears "I am turning left" from the emergency vehicle (like this is the case in Japan), it is hard to know what to do to help the emergency

vehicle as it is hard to know where is the vehicle. Getting information about the emergency vehicle location in the car could reduce this delay to 0 for all the people who have our system as the location of the emergency vehicle will be shown in the car navigation system.

Another important point we highlighted is that more than 40% of the drivers are not always sure about what they should do to help the emergency vehicles. We believe that knowing where the emergency vehicle wants to go would help them a lot to know what to do to help the emergency vehicles. Additionally, giving specific directions would even better help the drivers to make the right decision.

Also, the respondents react positively to the system. The majority agreed on all points that this kind of system could help them.

- not losing time looking for the emergency vehicle (75%)
- not losing concentration looking for the emergency vehicle (50%)
- not losing time finding out what to do to facilitate the way to the emergency vehicle (63%)
- not losing concentration finding out what to do to facilitate the way to the emergency vehicle (52%)

And finally, almost 80% of the respondents think that this kind of system can help them to make the right decision to help the emergency vehicles.

This is consistent with the fact that 90% of the respondents told that they will take into consideration the guidance given by such system.

b) Emergency vehicle drivers

At the Kanagawa Prefecture Ward Office, we had the chance to get feedback from a fire-truck driver who used to be an ambulance driver. He confirmed his interest in the system and believes that just the fact to have the location of the surrounding emergency vehicles on the drivers car navigation would already be a huge improvement to facilitate the emergency vehicle displacement.

One of his main concern as an emergency vehicle driver are the continuous accelerations/decelerations he has to perform because of the other vehicles obstructing the way. He believes that a system that control the traffic could help to have a smoother drive by reducing the number of vehicles on the way the emergency vehicles is going. Most of the drivers in front of an ambulance think that they can just keep driving like that if they are going in the same way. They often do not realize that the ambulance could drive faster if they park on the side and let the ambulance go first. Emergency drivers are often slowed down and frustrated because of that.

It appeared during the interview that ambulances are often slowed down at traffic light intersections because even if the emergency vehicles are allowed to drive at the red traffic light, the normal vehicles are not. So if normal vehicles are in front of the emergency vehicles at a red traffic light, the emergency vehicle is blocked. Automatically turning the light to green when an emergency vehicle arrives would help the emergency vehicle, but this is in application only for the police cars in Japan (and is expected to be extended to the ambulances). Controlling the traffic to prevent vehicles to go on the way the emergency vehicle is taken would help to reduce this problem.

The interview has been confirmed by the answers to the questionnaire for the emergency vehicle drivers. 60% of the 51 emergency vehicle drivers who answered to the survey said that most of the people try to facilitate the way to the emergency vehicles. 33% think however that only some of them try to facilitate the way to the emergency vehicles. As 90% of the normal vehicle drivers said that they try to facilitate the way, we can suppose that they do not always success in facilitating the way to the emergency vehicles, so one third of the emergency vehicles think that they only “sometimes” try to facilitate the way. This is consistent with the fact that 33% of the emergency vehicle drivers told that the other drivers often make the wrong decision when they try to help so it slows them down.

35% of the vehicles in front of the emergency vehicles wait for the very last moment (when the emergency vehicle is within 30 meters behind) to make a move. This explains the succession of accelerations and decelerations the emergency vehicle drivers have to do.

Finally, 70% of the emergency vehicle drivers are interested in the system; 94% including the ones who do not have a special feeling about the system.

III.3 System specification

As explained above, our system is an ITS Application that relies on wireless vehicular communication.

We call this application “Traffic Control for Emergency Vehicles” and refer to it as TC4EV. Figure III.1 gives an overview of the different components and its connections:

The main part is the block “ITS Applications”, which is a logical component that has access to the car navigation system (GPS, maps/routes, LCD display...), the vehicle dynamics (directly or through the car navigation system) and the wireless interface to

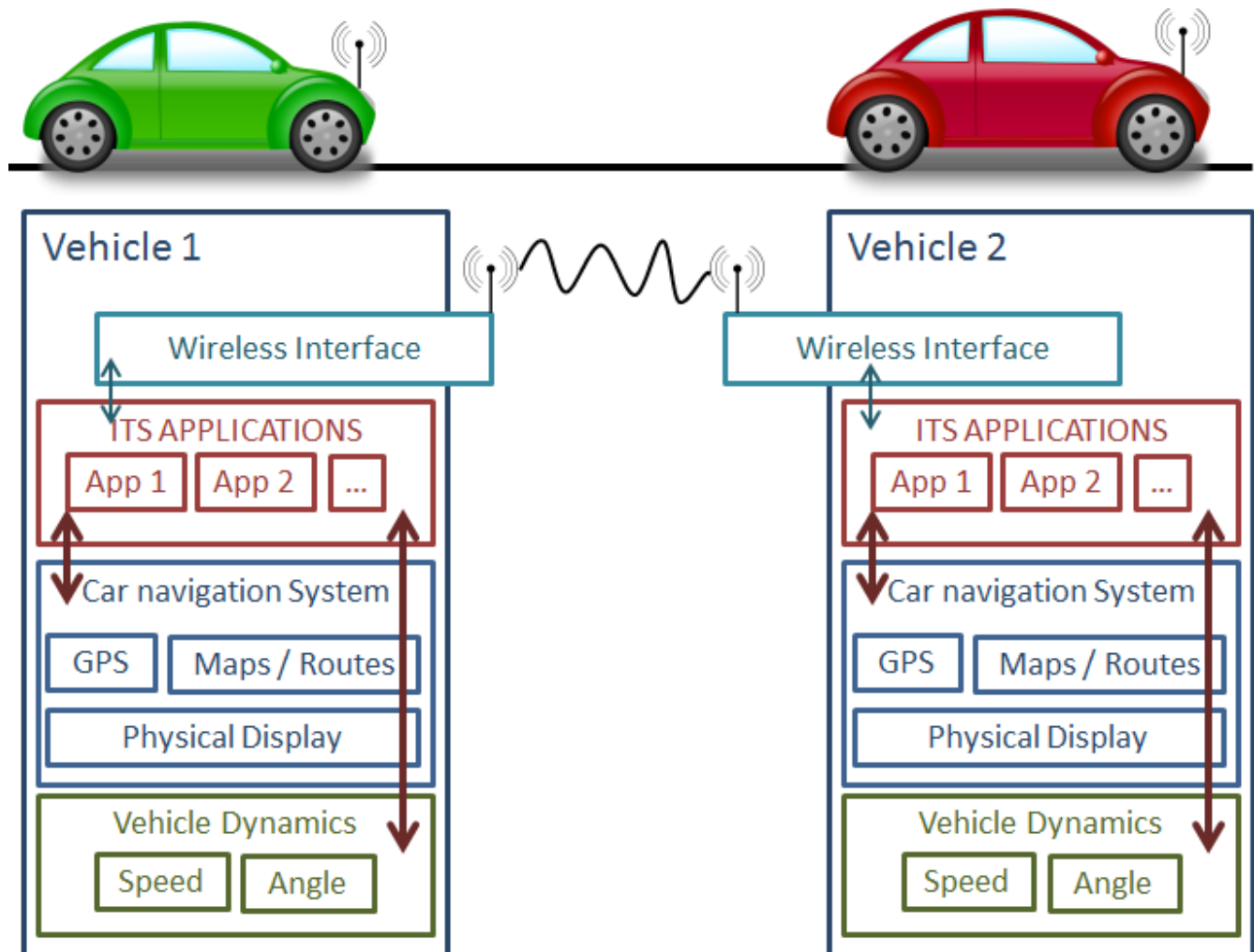


Figure III.1: Vehicle-to-vehicle communication components

communicate with other vehicles or infrastructure.

There can be many different ITS applications, like the ones introduced in the previous chapters. In this study, we will focus on the one we propose: TC4EV.

Depending on the complexity of the application, more or less components may be needed. In our case, we need to know the following information in the emergency vehicle:

- the location (longitude, latitude)
- the velocity and direction
- the identification of the next roads it is going to take (according to the car navigation system).
- the priority of the current emergency

For the surrounding vehicles, we need to have access to:

- the current location
- the planned route

The emergency vehicle broadcasts an emergency message through the vehicular wireless communication channel. For our application, we do not need a very frequent broadcasting of the message, to the contrary of other applications such as collision avoidance or adaptive cruise control, where the data as too be as recent as possible to be efficient. In our case, broadcasting every second is sufficient to inform the surrounding vehicles with fairly correct information.

The emergency message contains its current location, velocity and direction, and the route it plans to take. Other vehicles equipped with the vehicular wireless communication and the TC4EV application receive data from the emergency vehicle and display relevant information to the driver.

We assume that the drivers follow the instructions given by the car navigation system, which is a reasonable assumption according to the questionnaire we have done.

Figure III.2 schematizes the communication between an ambulance and a vehicle in case of the TC4EV application.

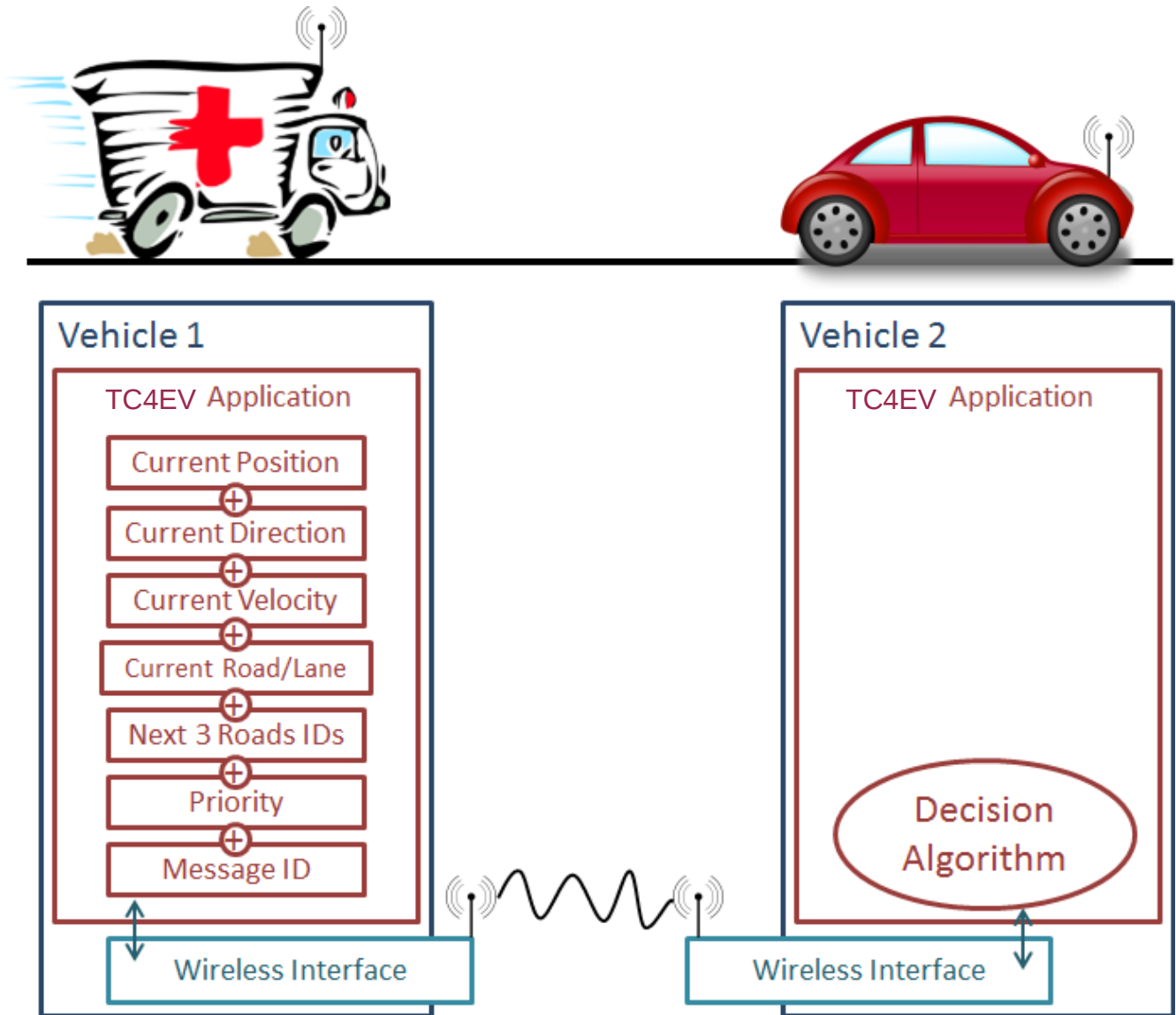


Figure III.2: Schematic representation of the TC4EV application

The priority has to be set by the emergency vehicle driver manually, depending on the importance of time matters in his current operation.

If we consider an ambulance carrying an injured person back to the hospital, if this person needs urgent medical support at the hospital otherwise he or she might perish before arriving, the priority will be set as its maximum. If instead the injured person life is not endangered, the priority can be set lower.

The reason why a priority is needed is to give priority to a given emergency vehicle over another one for when several emergency vehicles reach the same place but for different

operations. In this case, we can give priority to the one with higher priority. If both have the same priority, then we give priority to the closest one to us.

The different levels of priorities can be decided by the hospitals, the police stations and the fire stations together. The details of the priorities is beyond the scope of this study.

The message ID is a unique sequence identifying a given message. As the ambulance broadcasts the message every second, most of the vehicles will receive the same information multiple times. While periodic broadcasting is necessary to inform new arriving cars, it will be annoying for the drivers to be warned every second. So we need a mechanism to avoid multiple identical notifications, and this message ID field does the job. The emergency vehicle will generate a new message ID every time it takes another street (as the “next roads” are different) or every time it rides 500 meters (to update its location in case of a highway when the next roads are not changing for a while), or finally after 10 seconds if the above conditions above have not been fulfilled.

The next 3 roads IDs are used to determine the route of the emergency vehicles. Knowing the current road, and only 1 road after does not indicate what direction is going to take the emergency vehicle, as shown in Figure III.3.

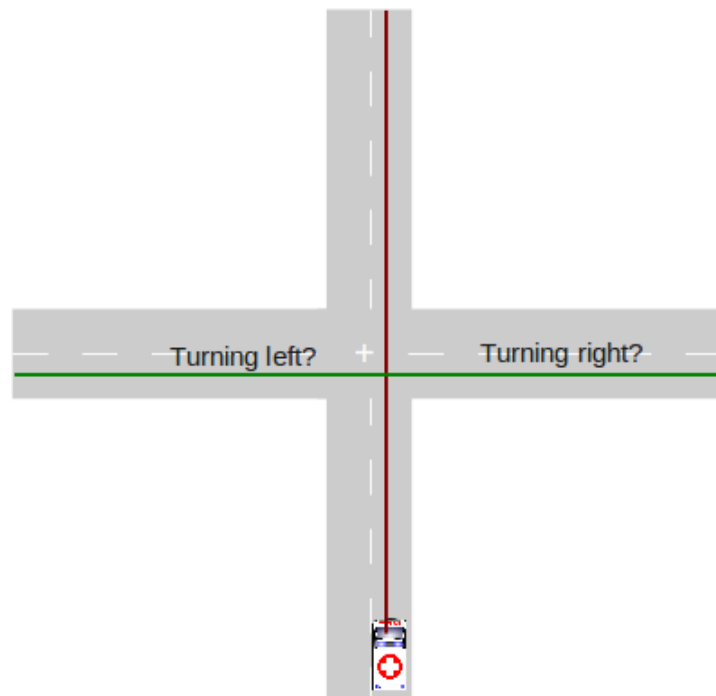


Figure III.3: Lack of information if only 2 roads are known

The red line is the current road, and the green line is the next road the ambulance is going to take. With only that information, we know that the ambulance is going to turn, but we do not know if it is right or left.

By knowing only two roads, we do not know which direction the emergency vehicle will take at the intersection.

By knowing three roads, we know where is turning the emergency vehicle. We can also extend to 4 roads if we want to know the next turn, and so on.

Figure III.4 shows the benefits of knowing 3 roads instead of 2.

In this case, we know at least that the emergency vehicle is going to turn right at the next intersection. So if we are arriving from the left road and planning to go straight on the right side (on the road with a green line that is going to take the emergency vehicle), we know the emergency vehicle is also planning to go there so we can give a notification to the driver that an emergency vehicle is arriving and that the driver should wait at the intersection until the emergency vehicle passes, or find a new route to its own destination that does not cross the emergency vehicle route (turning left or right at the first intersection in this case). This scenario is illustrated in Figure III.5.

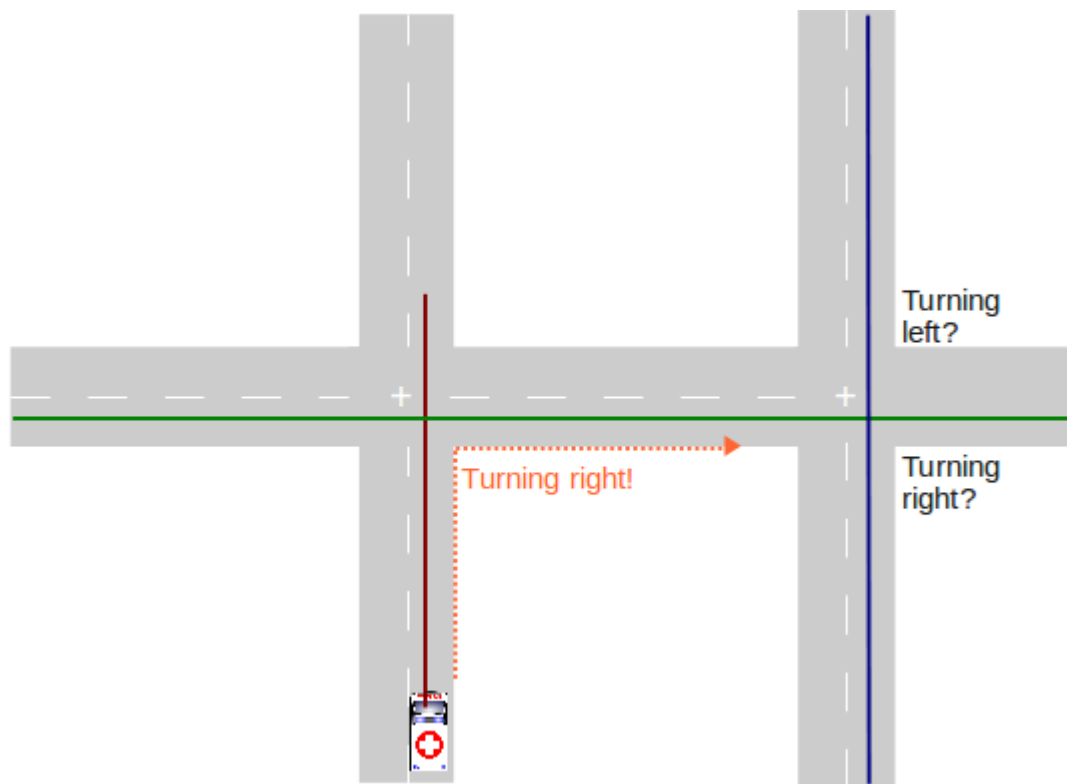


Figure III.4: Benefit of knowing at least three roads of the EV

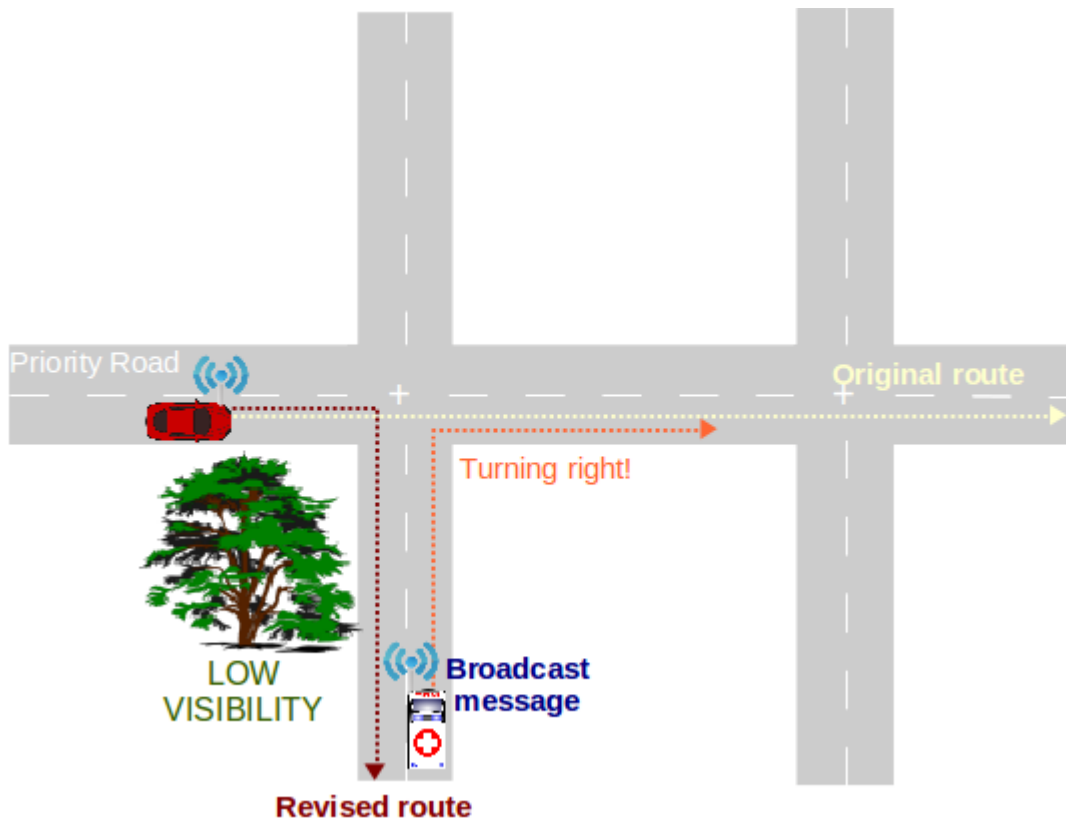


Figure III.5: Example of a revised route to prioritize the EV

The red car was planning to go straight in the priority road, but an emergency vehicle announced that it was going to take this road, in the same direction of the red car. Thus, the red car navigation system recalculates a route to reach its own destination that does not cross the emergency vehicle route. For example by turning right at the next intersection (and then probably left to catch up the initial plan).

Additionally, the possibility of stopping at the intersection instead of changing the plan can be displayed to the driver on its car navigation system. In this case, if possible, the red car driver can choose to stop at the intersection and wait for the emergency vehicle to pass first.

Both options should be presented to the driver.

The algorithms to generate the message and upon reception of an emergency message will now be presented.

On the emergency vehicle side: generate a new message (and broadcast it).

Skeleton of the algorithm on the sender side:

```
# Generate a new message ID if needed
IF (distance(previous_location, current_location) > 500)
  OR (previous_time - current_time > 10)
  OR (previous_road ≠ current_road)
  OR (previous_lane ≠ current_lane)
THEN message_id ← Generate_New_Message_ID()
END IF
# Prepare the packet
packet ← New_TC4EV_Packet(message_id, priority, current_location,
                          current_velocity, current_direction,
                          current_road, current_lane,
                          next_road_1, next_road_2, next_road_3)
Send_Packet(packet)
```

This simple algorithm is called every second to broadcast the information to the surrounding vehicles.

The most important task is on the receivers side that we will now introduce.

To better understand the process on the receiving side, a state machine representation is introduced in Figure III.6.

When a vehicle receives a notification from an emergency vehicle, we see from this diagram that there are 5 possible outputs:

- Update display: if the message is not new or if we do not block the emergency vehicle, just update the emergency vehicle position and route on the map of the car navigation system
- Change route and inform the driver: if we are going in the future according to our traveling plan to intersect with the emergency vehicle, and if there is a way to avoid it by choosing another route, we choose that other route. We also propose to stop at the next intersection and let the emergency vehicle go first.
- Ask the driver to stop at the next intersection with the emergency vehicle: at the next intersection, the emergency vehicle will cross our path. Thus, we ask the driver to wait at that intersection that the emergency vehicle passes first before going. In the case the emergency vehicle is already behind the vehicle, do not “stop” at next intersection but let the emergency vehicle overtake you.

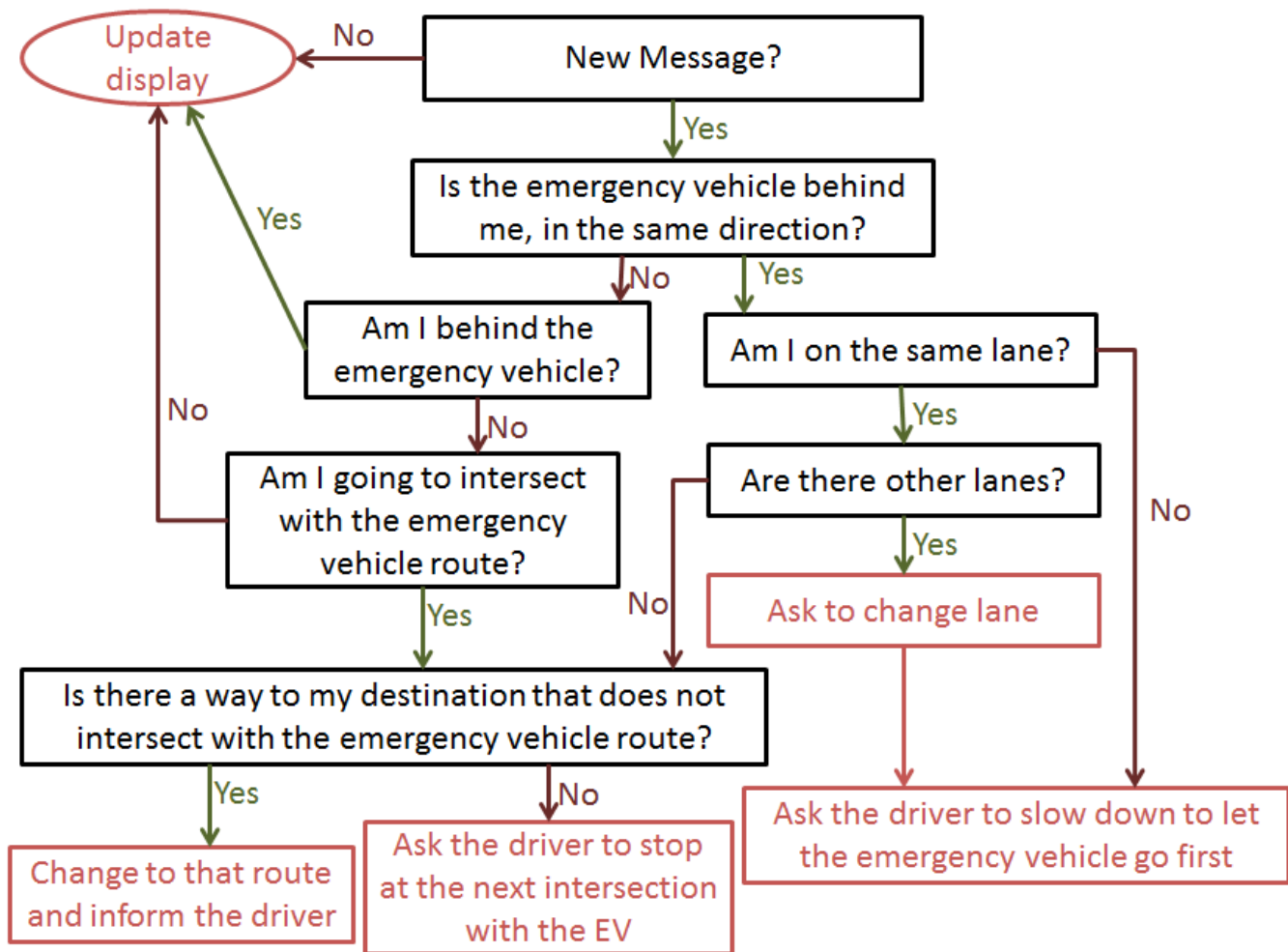


Figure III.6: State-machine representation of the decision algorithm

- Ask to change lane: if we are on the same lane of an emergency vehicle behind us and there are other lanes, we ask the driver to change lane to let the current lane free for the emergency vehicle.
- Ask the driver to slow down to let the emergency vehicle go first: if the emergency vehicle is behind us and that there are multiple lanes, ask to slow down into the lane not occupied by the emergency vehicle and let the emergency vehicle go first.

In order to validate that algorithm, we developed an application that implements such algorithm and simulate the message returned by the car navigation system.

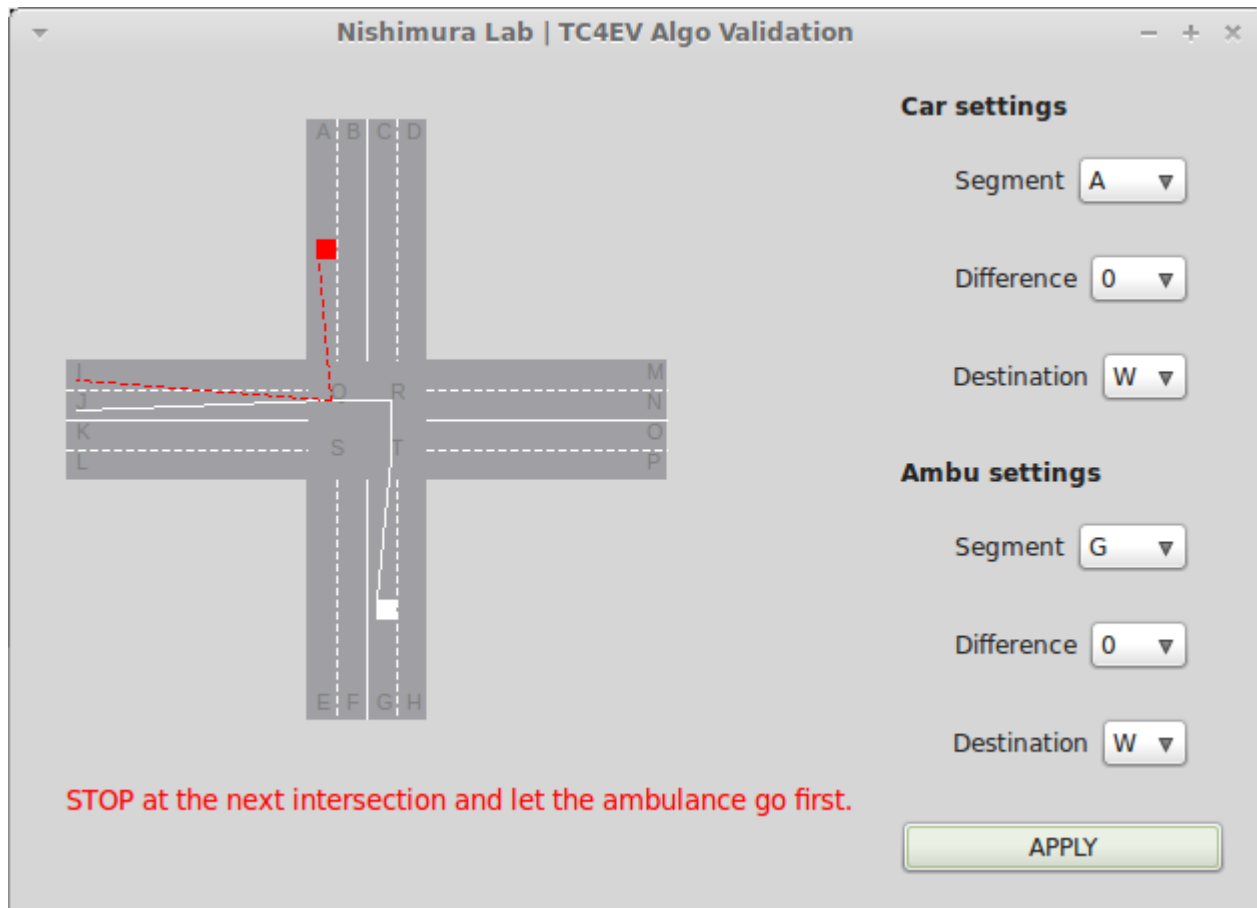


Figure III.7: Screenshot of the program to validate the decision algorithm

An intersection of two streets of two ways with 2 lanes each has been designed. The position of the emergency vehicle (an ambulance in the program) and the position of the car receiving information can be defined. The destination of the car and of the ambulance can also be defined. According to the situation, the information displayed on the car navigation system is displayed by the program.

In Figure III.7, we see on the left a schematic representation of the intersection. In red is represented the car and in white the ambulance. The car is in the segment A, in the middle (difference is 0) and its destination is West (W). The ambulance is in the middle of the segment G, going West also.

We see that in that case, the car navigation system advises the driver to “STOP at the next intersection and let the ambulance go first”, as wanted.

The full source code of this program is a about 500 lines long and only the most important lines that perform the decision are introduced here. The full source code of this program can be found in Appendix B.

The program is written in Python and the decision algorithm part has been heavily documented.

Decision algorithm part of algorithm_validation.py:

```
# =====
# = PERFORM THE ALGORITHM =
# =====
# distance of the first line of the car and the ambu
carFirstLineDist = dist(*carPath[0])
ambuFirstLineDist = dist(*ambuPath[0])
# If the car has already passed the intersection
# or is at the intersection
if len(carPathSeg)==1 or atIntersection(carSegment):
    # and the ambulance also
    if sameWay(carSegment, ambuSegment):
        # and even at the same segment
        if carSegment==ambuSegment:
            # if the car is ahead
            if carFirstLineDist < ambuFirstLineDist:
                # and can change lane, we ask to change lane
                if multipleLanes:
                    naviMessage = 'Please change lane, the ambulance is behind you'
                    if carDest==ambuDest:
                        naviMessage += '\nIf not possible, please change your way to %s'\
                                % redirect(carSegment,carDest)
                # else if we have the same destination, change the car destination
            elif carDest==ambuDest:
                naviMessage = 'Please change your way to %s' %\
                        redirect(carSegment,carDest)
            else: # else there is nothing we can do except continuing
                naviMessage = 'Ambulance is behind you. Stick to your plan.'
        # else, we are in the same way, but not the same segment
    elif carFirstLineDist < ambuFirstLineDist:
        # if we are ahead, give priority to the ambulance
        naviMessage = 'Stay in this lane and \
                slow down to let the ambulance go first.'
    # else, we should see it, no need to worry the driver more
# else, we are not in the same way
```

```

elif ambuDest==carDest and len(carPathSeg) < len(ambuPathSeg):
    # if the car is coming this way, let the driver know it
    naviMessage = 'Ambulance will come this way.'
    # else, there is nothing special to do
    # (we are at the end segment and the ambu is not coming)
# Else, if we are in the same way (but not the last one)
elif sameWay(carSegment, ambuSegment):
    # and the same segment
    if ambuSegment==carSegment:
        # if the car is ahead
        if carFirstLineDist < ambuFirstLineDist:
            # and can change lane, advise the driver to do so
            if multipleLanes:
                naviMessage = 'Please change lane, slow down, \
                    and let the ambulance go first.'
            # else, if we have the same destination,
            # change our way to the destination
            elif carDest==ambuDest:
                naviMessage = 'Please change your way to %s'\
                    % redirect(carSegment,carDest)
            else: # else there is nothing we can do except continuing
                naviMessage = 'Ambulance is behind you. Stick to your plan.'
        # else, we are in the same way but not the same segment
    elif carFirstLineDist < ambuFirstLineDist:
        #if we are ahead, let the driver know to stay in this lane but to slow down
        naviMessage='Stay in this lane and slow down to let the ambulance go first.'
        # else there is nothing special to do
        # (same way, different segment, behind the ambu)
# Else, if the ambu is at its last segment
# or at the intersection before its last segment
# there is nothing special
elif len(ambuPathSeg)==1 or atIntersection(ambuSegment):
    naviMessage = 'DISPLAY INFORMATION ONLY (2)'
# else, we are in different ways and both going to cross the intersection
# check if our routes intersect
elif routesIntersect(ambuPathSeg, carPathSeg):
    naviMessage = 'STOP at the next intersection and let the ambulance go first.'
# else, we are in different ways and are not going to cross our routes,
# just display info
else:
    naviMessage = 'DISPLAY INFORMATION ONLY (3)'
self.message.emit(naviMessage)

```

The main scenarios will be now introduced.

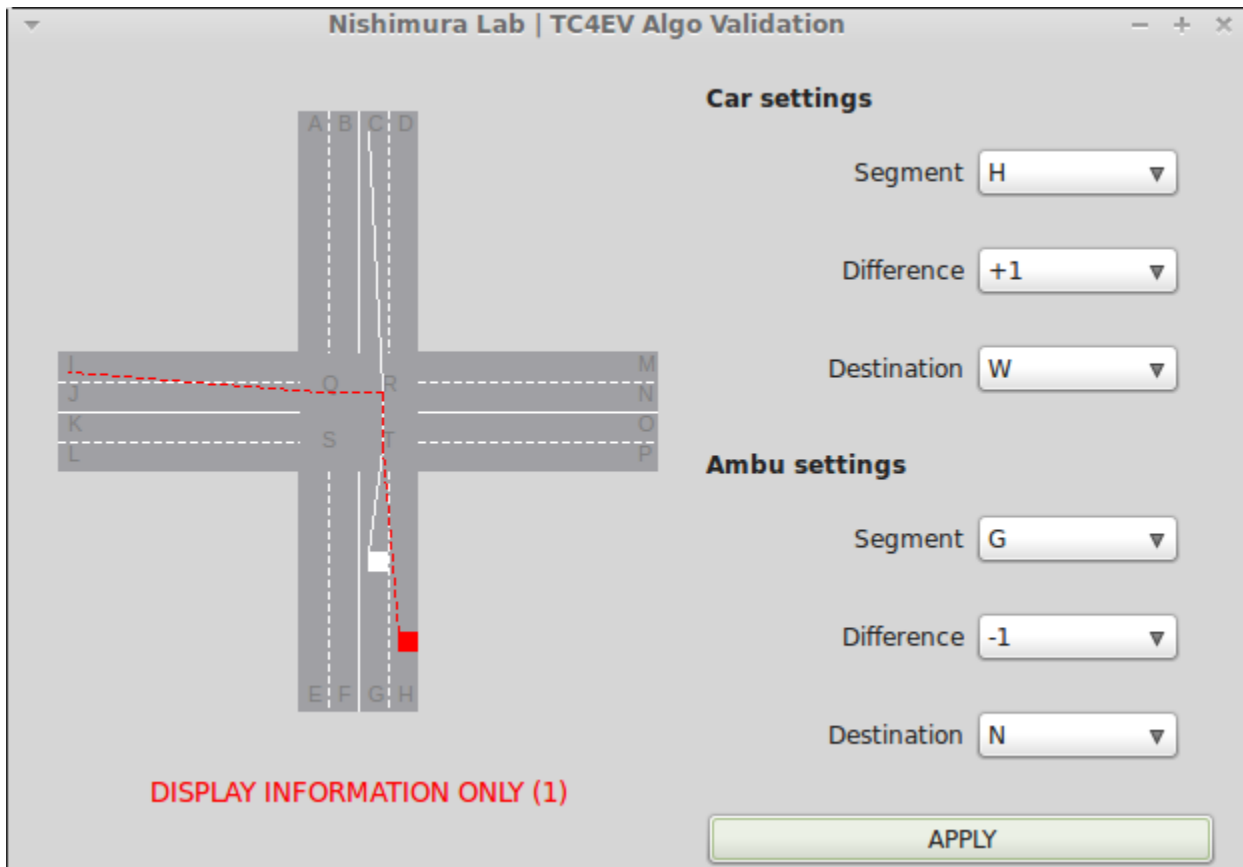


Figure III.8: TC4EV example when the car is behind the EV

First, an example when the car is behind the emergency vehicle. In such case, we do not need to ask anything to the driver but only update and display the information on the car navigation system (by “display information” we mean display where is the emergency vehicle on the map and the route it is going to take), as shown in Figure III.8.

Then, the same situation, but if the emergency vehicle is behind the car is shown in Figure III.9.

As expected, the driver is asked to stay in that lane and give priority to the oncoming emergency vehicle.

If the car was in the same lane as the ambulance, the driver would have been asked to change lane and give priority to the oncoming emergency vehicle.

In addition, if that is not possible, the car navigation system advises the driver to turn right at the next intersection (to go North) as shown in Figure III.10.

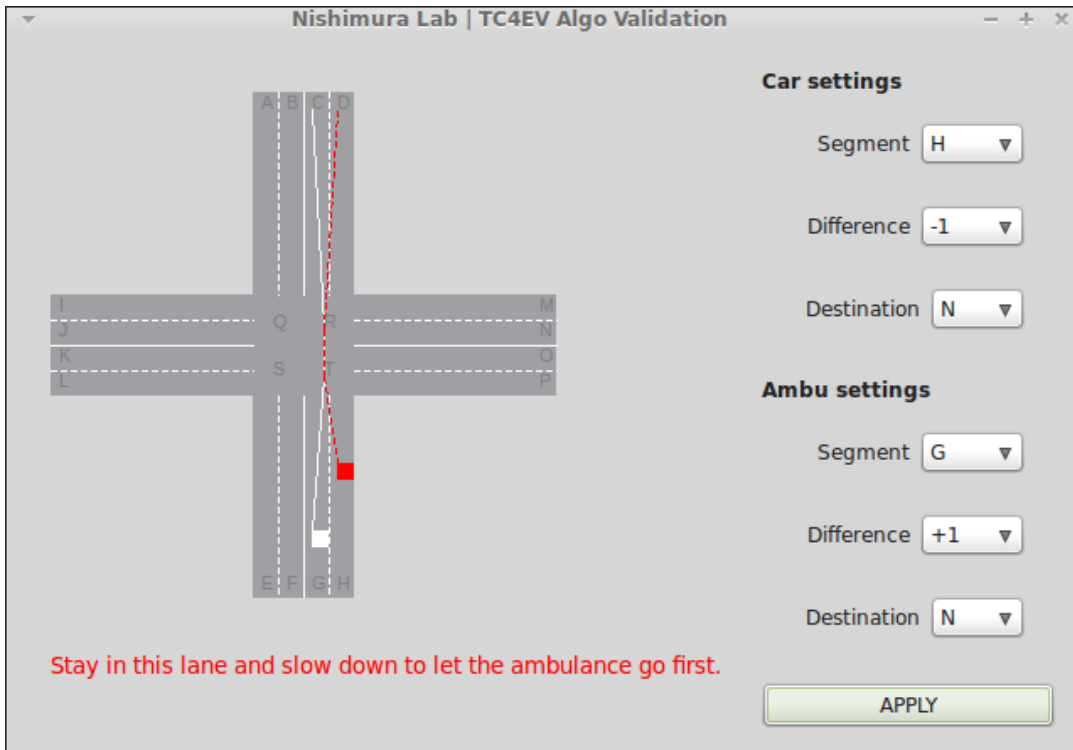


Figure III.9: TC4EV example when the EV is behind the car (1)

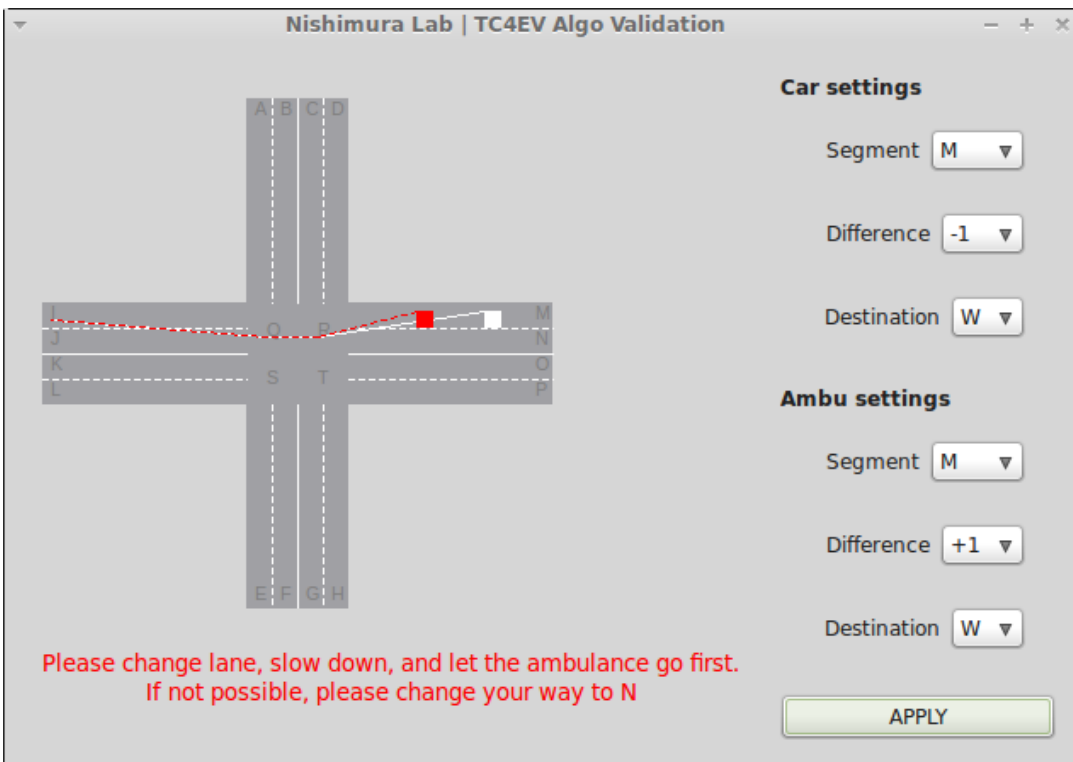


Figure III.10: TC4EV example when the EV is behind the car on the same lane

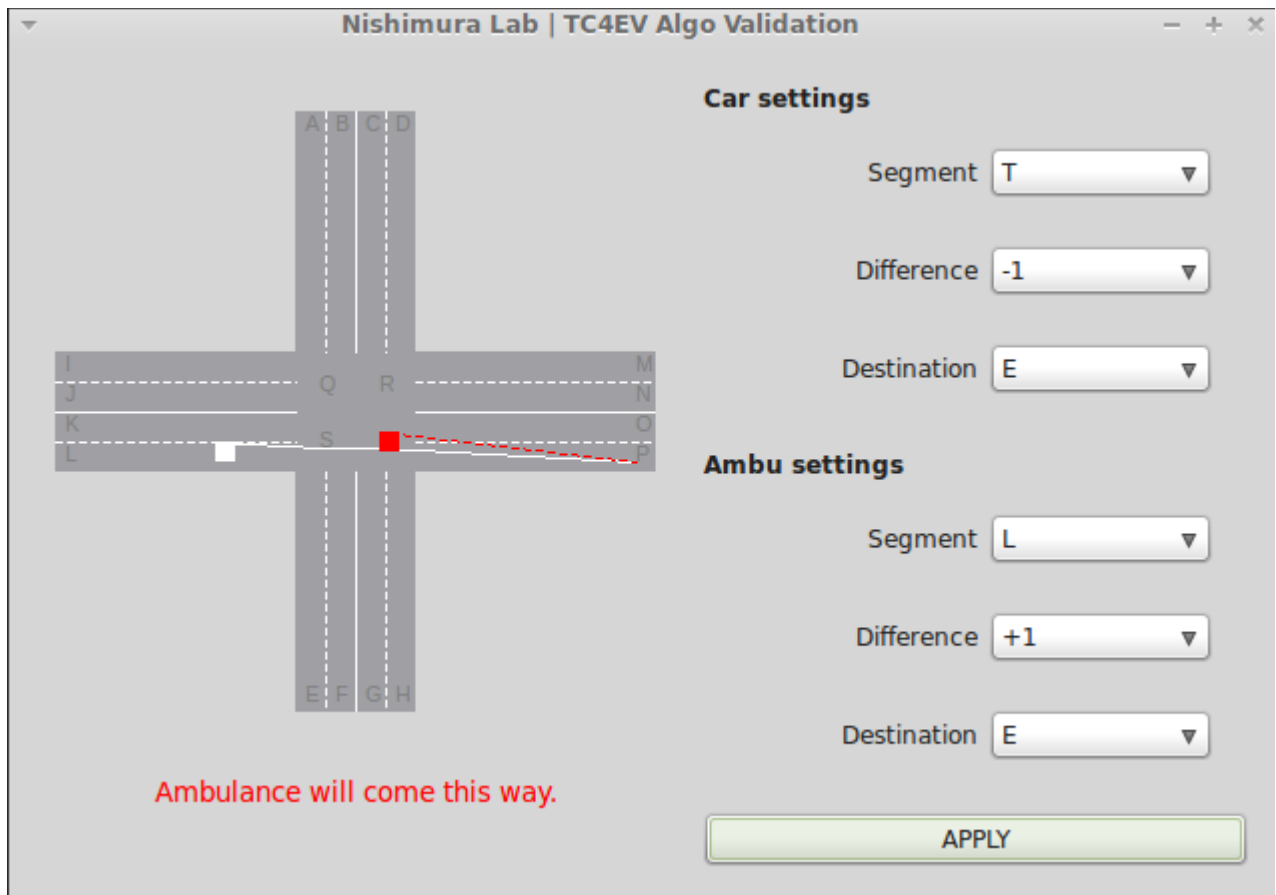


Figure III.11: TC4EV example when the car already passed the intersection

Finally, if we are already engaged in the intersection or even after the intersection, and the ambulance is coming to our way, we inform the driver about it. More instructions will be available when the emergency vehicle will also pass the intersection but for the moment we can only warn the driver that the ambulance is also coming here, like shown in Figure III.11.

III.4 Responsibility matters

In case of an accident that occurred because a driver, who, for instance, changed lane as the car navigation system suggested (from the TC4EV application), without paying too much attention, and bumped into a motorbike that was coming from the blind spot, one can ask if the TC4EV application has some part of responsibility or not. And the answer is that it does not, given that the TC4EV application is part of the car navigation

system (the displayed information) and so on obey the same responsibility matters, which means is not responsible for the actions of the driver.

The car navigation system (and the TC4EV application) does not control directly the mechanics of the vehicle (brake, wheel...), at least for the moment (maybe in the future). Therefore, the driver is the one who has the final decision to follow or not the directions given by the car navigation system and it is his/her own responsibility to change lane or perform any action.

For example, the car navigation system sometimes makes mistakes, like saying “in 100 meters, turn right” where there is no intersection. In such case, if the driver turns right and falls into the ditch, it is not the car navigation system's responsibility but the driver's one.

III.5 Technology alternative

Some applications, like collisions avoidance, require a very short transmission time (100 ms). However, for our application, we can envisage the acceptance of a few seconds delay. After all, if the data is received within a reasonable time, there is still a good chance that it is early enough to facilitate the way to the emergency vehicle in advance. According to a research by I. Lequerica et al. in 2010 about improving the vehicular communications by using 3G capabilities, end-to-end delays of 500 ms can be achieved in recent 3G networks^[12].

In such case, it is conceivable to use the 3G network to support our application. The main advantage of doing so is that all the mobile phones connected to internet through the 3G network can provide the capability of TC4EV to an existing car navigation system. Thus, the deployment of the system can be much faster than waiting for the manufacturers to implement the IEEE 802.11p interface.

However, there might be a compatibility problem to handle, as the 3G connected vehicles will receive information through a centralized internet server, whereas the VANET technology does not rely on any distant infrastructure (because it is ad-hoc). So in order to receive the information on the mobile phones of the drivers, it requires the

emergency vehicle to send the information through the 3G network. Thus, the emergency vehicles would need both interfaces (IEEE 802.11p and 3G). It is not convenient but still conceivable.

Nevertheless, it can help to deploy quickly the application but is not a sustainable solution as data transmission on the cellular networks is not reliable and delays of more than 10 seconds might occur (transmitting voice allows network errors which are ignored and the conversation is still understandable, but such errors are not acceptable for data transmission, as the information might be erroneous).

We will now evaluate the benefits of our system in the next part.

IV. Evaluation of the system

To evaluate our system, we need to incorporate two different types of simulations together:

- network simulation (to simulate the wireless communication)
- vehicular simulation (for the vehicles displacements)

Mr. A. Hassan did a “VANET Simulators Comparison” for his Master’s Thesis in Electrical Engineering in 2009^[26]. According to his study, the simulator that corresponds the most to our needs seems to be NCTUns.

NCTUns defines itself as follows: “With more than nine years of development, NCTUns now is a professional tool that competes with commercial software. In addition, with its unique kernel-reentering simulation methodology, NCTUns offers many unique advantages that cannot be easily achieved by traditional network simulators and emulators. NCTUns is a valuable tool for network planning, network research, and network education”.

However, victim of its success, NCTUns became commercial and renamed EstiNet in 2010. The old version of NCTUns has been removed from their website and the forum where developers of NCTUns actively helped the users to use the software has been closed and erased, and they do not provide any support anymore. We have been using it for a few months and always found answers on the support section of NCTUns but since it became commercial, there is no way to get help if you do not get a license of the commercial version. Furthermore, as the commercial version is recent, they were in standby during the migration so it was impossible to get any support. As a result, we searched for alternatives to NCTUns to continue my simulations.

Since Mr. Hassan's study, a new addition for the popular “network simulator 3” has been released. In December 2010, H. Arbabi and M. C. Weigle published a research entitled “Highway Mobility and Vehicular Ad-Hoc Networks in ns-3”^[27].

ns-3 (Network Simulator 3) is a “discrete-event network simulator for the Internet systems, targeted primarily for research and educational use”. It is a “free software,

licensed under the GNU GPLv2 license, and is publicly available for research, development and use". It is written in C++ and Python and intended as a replacement for the popular ns-2 simulator. ns-3 promises to be a more efficient and more accurate simulator than its predecessor (especially for wireless protocols).

The additions by H. Arbabi and M. C. Weigle allow the simulation of highways in ns-3. Unfortunately, intersections cannot be simulated with these additions, but we can still perform simulations on highway. We will use it to evaluate our system in highway conditions.

IV.1 Highway Simulation Environment

The highway simulation is based on the Intelligent Driver Model (IDM) by Treiber et al. (2000)^[28] and the MOBIL lane-change model by Treiber and Helbing (2002).

In traffic flow modeling, the intelligent driver model (IDM) is a time-continuous car-following model for the simulation of freeway and urban traffic. It was developed by Treiber, Hennecke and Helbing in 2000 to improve upon results provided with other "intelligent" driver models such as Gipps' Model, which loose realistic properties in the deterministic limit.

a) Intelligent Driver Model

As a car-following model, the IDM describes the dynamics of the positions and velocities of single vehicles. For vehicle α , x_α denotes its position at time t , and v_α its velocity. Furthermore, l_α gives the length of the vehicle. To simplify notation, we define the *net distance* $s_\alpha := x_{\alpha-1} - x_\alpha - l_{\alpha-1}$, where $\alpha-1$ refers to the vehicle directly in front of vehicle α , and the *velocity difference*, or *approaching rate*, $\Delta v_\alpha := v_\alpha - v_{\alpha-1}$. For a simplified version of the model, the dynamics of vehicle α are then described by the following two ordinary differential equations:

$$\dot{x}_\alpha = \frac{dx_\alpha}{dt} = v_\alpha \quad \dot{v}_\alpha = \frac{dv_\alpha}{dt} = \alpha \left(1 - \left(\frac{v_\alpha}{v_0} \right)^\delta - \left(\frac{s^*(v_\alpha, \Delta v_\alpha)}{s_\alpha} \right)^2 \right)$$

$$\text{with } s^*(v_\alpha, \Delta v_\alpha) = s_0 + v_\alpha T + \frac{v_\alpha \Delta v_\alpha}{2\sqrt{ab}}$$

v_0 , s_0 , T , a , and b are model parameters which have the following meaning:

- *desired velocity* v_0 : the velocity the vehicle would drive at in free traffic
- *minimum spacing* s_0 : a minimum net distance that is kept even at a complete stand-still in a traffic jam
- *desired time headway* T : the desired time headway to the vehicle in front
- *acceleration* a
- *comfortable braking deceleration* b

The exponent δ is usually set to 4 and set to 4 in our simulations.

The acceleration of vehicle α can be separated into a free road term and an interaction term:

$$(\dot{v}_\alpha)_{free} = \alpha \left(1 - \left(\frac{v_\alpha}{v_0} \right)^\delta \right)$$

$$(\dot{v}_\alpha)_{inte} = -\alpha \left(\frac{s^*(v_\alpha, \Delta v_\alpha)}{s_\alpha} \right)^2 = -\alpha \left(\frac{s_0 + v_\alpha T}{s_\alpha} + \frac{v_\alpha \times \Delta v_\alpha}{2\sqrt{ab} s_\alpha} \right)^2$$

- *Free road behavior*: On a free road, the distance to the leading vehicle s_α is large and the vehicle's acceleration is dominated by the free road term, which is approximately equal to a for low velocities and vanishes as v_α approaches v_0 . Therefore, a single vehicle on a free road will asymptotically approach its desired velocity v_0 .
- *Behavior at high approaching rates*: For large velocity differences, the interaction term is governed by $-\alpha (v_\alpha \times \Delta v_\alpha)^2 / (2\sqrt{ab} s_\alpha) = -(v_\alpha \times \Delta v_\alpha)^2 / (4b s_\alpha^2)$.

This leads to a driving behavior that compensates velocity differences while trying not to brake much harder than the comfortable braking deceleration b .

- *Behavior at small net distances:* For negligible velocity differences and small net distances, the interaction term is approximately equal to $-\alpha(s_0 + v_\alpha T)^2 / s_\alpha^2$, which resembles a simple repulsive force such that small net distances are quickly enlarged towards an equilibrium net distance.

b) Lane-change model

The MOBIL lane-change model by M. Treiber and D. Helbing allows lane change, if:

- the potential new target lane is more attractive, i.e., the "incentive criterion" is satisfied,
- and the change can be performed safely, i.e., the "safety criterion" is satisfied.

The safety criterion is satisfied if the IDM braking deceleration $acc' = acc'_{IDM}$ imposed on the back vehicle B' of the target lane after a possible change does not exceed a certain limit b_{save} , this means, the safety criterion $acc' (B') > - b_{save}$ is satisfied. In this formula, the dash of the acceleration $acc' = acc'_{IDM}$ stands for "after a possible change", while the dash of the back vehicle label B' stands for "on the target lane".

To assess the incentive criterion, we weight the own advantage on the target lane, measured by the increased acceleration (or reduced braking deceleration), against the disadvantage imposed to other drivers, again measured by the decrease acceleration or increased braking deceleration for these drivers. Since we tend to be egoistic, we weight the disadvantage imposed on other drivers with a politeness factor p whose values are typically less than 1, resulting in following incentive criterion:

$$acc' (M') - acc (M) > p [acc (B) + acc (B') - acc' (B) - acc' (B')] + a_{thr}$$

As above, acc mean the actual IDM accelerations while acc' mean the accelerations after a possible change. The car labels M and M' mean "Me" before and after a possible change, respectively, while B and B' mean the back vehicle before and after a possible

change, respectively.

The own advantage is measured by "my" acceleration difference $acc'(M') - acc(M)$ after the change, compared to the actual situation.

The combined disadvantage to the new and old back vehicles is given by the sum $[acc(B) + acc(B')]$ of the accelerations of both vehicles before the change, minus the acceleration sum $[acc'(B) + acc'(B')]$ of these vehicles after the change.

To avoid lane-change manoeuvres triggered by marginal advantages which can lead to frantic lane hopping, an additional lane-changing threshold a_{thr} has been added to the balance of the above equation.

c) Scenario and parameters of the simulation

The scenario is a highway where vehicles can be equipped with the VANET technology and the TC4EV application. The percentage of equipped vehicles can be determined. We consider two lanes and the possibility to change lane. We record the time for the emergency vehicle to cover 10 kilometers.

The following values are set to the different parameters of the lane change model:

- politeness factor $p = 0.5$ (default value for realistic behavior; will change lane if it is at its own advantage and not bothering other drivers)
- maximum safe deceleration $b_{save} = 4 \text{ m/s}^2$ (default value for realistic behavior)
- threshold $a_{thr} = 0.2 \text{ m/s}^2$ (default value for realistic behavior)

However, when a vehicle receives a TC4EV message asking to change lane, the politeness factor is lowered to -0.5 so that the drivers try to change lane (still respecting safety rules) even if it is at the cost of own disadvantages.

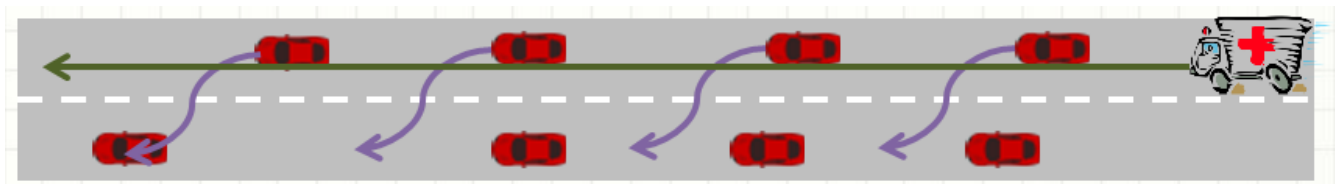


Figure IV.1: Highway simulation representation

If the cars do not have the TC4EV capabilities, they follow a normal driver model to give priority to the emergency vehicle by reducing the politeness factor to -0.5 when the emergency vehicle is behind the vehicle within X meters, X following a normal distribution of average value 120 (meters), minimum 20 and maximum 220. It simulates drivers on a highway from people who do not really want to give priority to the emergency vehicle unless they are really blocking the vehicle (20 meters) to people who start changing lane from a long distance ahead (220 meters).

For the intelligent driver model, the following values of the parameters have been applied:

- desired velocity: average of 105 km/h (normally distributed)
- safe time headway has been set to 1.5 sec (default value for realistic behavior)
- maximum acceleration: 1 m/s² (default value for realistic behavior)
- desired deceleration: 3 m/s² (default value for realistic behavior)
- jam distance: 2 meters
- vehicle length: 5 meters

The emergency vehicle, when in emergency situation, sets its desired deceleration to 6 m/s² to allow more aggressive driving behavior (because the emergency vehicle driver is in a hurry). Its desired velocity has also be increased to a little less than 165 km/h that it will try to reach if there are no cars in the way (free road).

Simulations will be performed with different number of vehicles per kilometers: 5, 10 and 20.

For each case, the TC4EV application will be evaluated depending on the percentage of vehicles equipped, from 0% (no vehicle has the application) to 100% (all the vehicles have the application).

Due to the presence of random variables in our simulations (for speed, initial distances, drivers behavior), each simulation (with given number of vehicles per kilometer and given number of TC4EV-equipped vehicles percentage) will be performed a minimum of 100 times.

d) Highway Simulations GUI

ns-3 is a command-line tool. It means it does not have any graphical interface to set up the parameters of the simulation. Everything has to be done by defining scenarios in Python/C++ and calling it with the right parameters through a command-line.

One simulation with fixed parameters at a time, and only textual output (log, result of the simulation). There is no built-in interface to allow batch simulations and data analyzing.

Therefore, we developed a graphical user interface (GUI) on the top of ns-3.

The interaction with the different components is shown in Figure IV.2.

The TC4EV Application has been coded inside the ns-3 code (which is open source) and ns-3 has been recompiled with the additional scenarios introduced previously.

The GUI can launch batch simulations and perform data analysis on the results.

Furthermore, it interacts with the Gnuplot portable command-line driven graphing utility.

Gnuplot is copyrighted but freely distributed. It was originally created to allow scientists

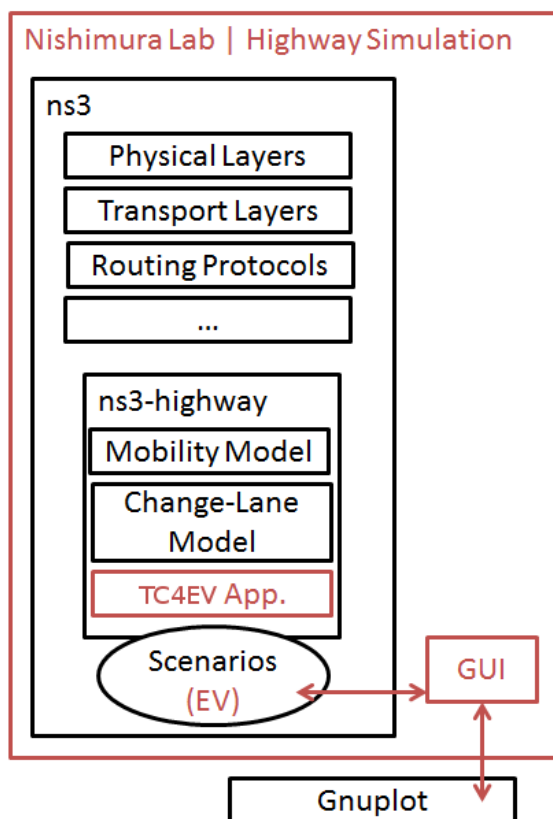


Figure IV.2: Highway Simulation GUI Components

and students to visualize mathematical functions and data interactively. It has grown to support many non-interactive uses and is used as a plotting engine by third-party applications like Octave... and our GUI.

Gnuplot allows to plot graphs from the output. We will plot the velocity and acceleration of the emergency vehicle over the time for each simulation and the time to reach the destination of the emergency vehicle depending on the penetration percentage of TC4EV for a given number of vehicles per second (automatically from all the simulations done through the GUI).

The GUI has the following features:

- Graphical set up the parameters (velocity, vehicles per kilometer, etc...)
- Selection of the scenario to simulate
- Selection of the gap between the different penetration percentage of TC4EV to evaluate (for example if set to 5, the GUI will batch simulations from 0% to 100% with a gap of 5% between the simulations: 0, 5, 10, ..., 95, 100)
- Data analysis (with criteria selection: consider the simulations with a given number of vehicles per kilometer, and ignore the result if a point has been obtained from less than a given number of simulations)
- Plot output to visualize a given result (time to reach destination depending on TC4EV penetration percentage in our case)
- Possibility to keep running simulations until manual stop

The first tab of the GUI is for running the simulations and can be seen in Figure IV.3.

On the top left part, we can set up general parameters, such as selecting the scenario to run, the output path, the number of lanes or the speed limit of the road. The VANET parameters can also be selected. Only the transmission power can be changed, all the other parameters being set to the official protocol values. 21.50 dBm is also the official transmission power value but we allowed to change it here for testing purpose.

On the top right part, the traffic settings can be set up, like the average distance between vehicles, their speed, desired speed (Intelligent Driver Model) and the ambulance specific parameters. The batch settings can also be chosen to perform automatic batch simulations.

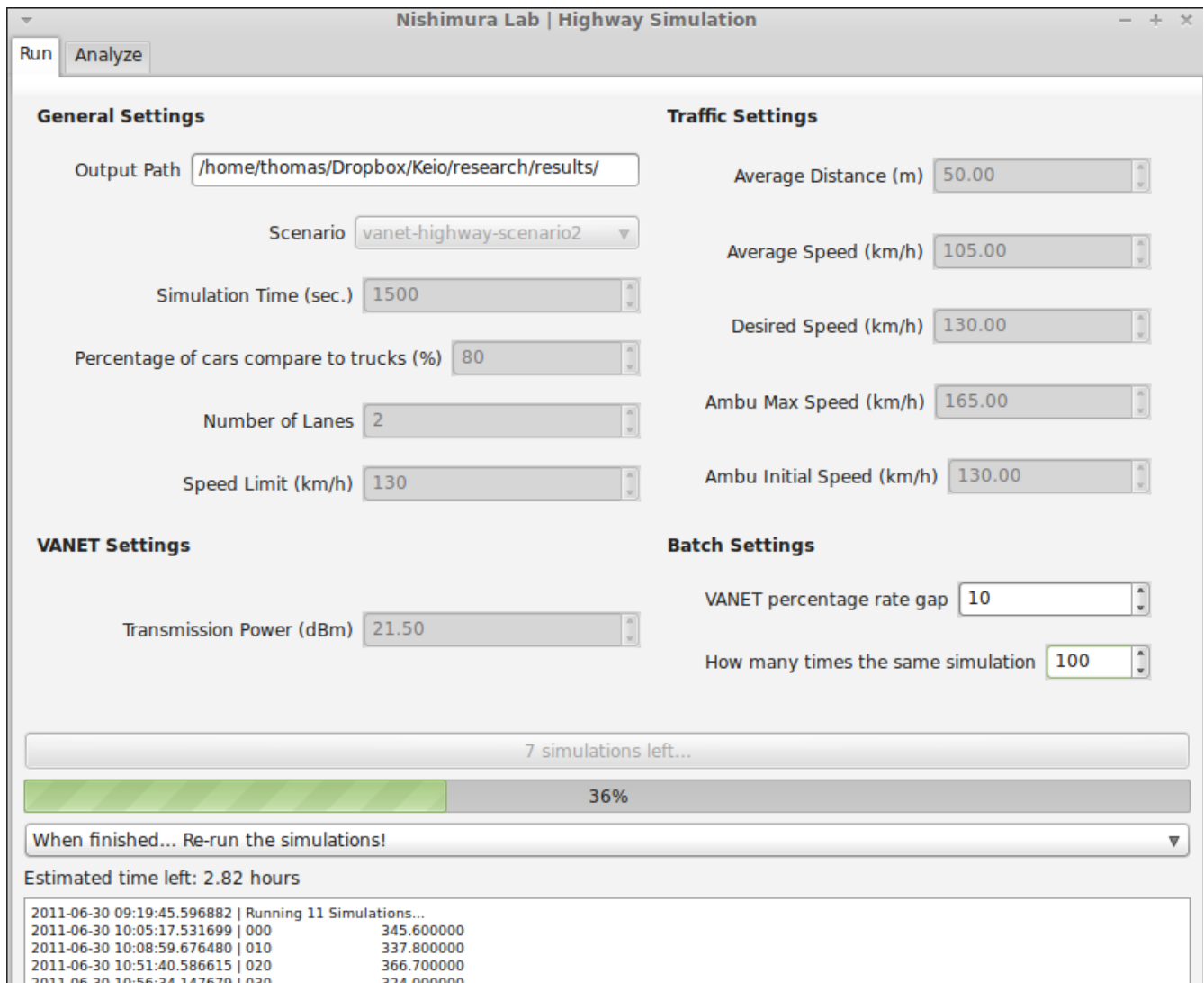


Figure IV.3: Highway Simulation GUI - Run

Then, the second tab allows to analyze the results of the simulations done through the first tab.

Giving the directory path to the results, the software can read the results and directly plot the output to the user. For example in the screenshot shown in Figure IV.4, one can choose to plot the graph for the scenario "vanet-highway-scenario2" where the average distance between cars is 100 meters and to invalidate the points where less than 30 simulations have been performed to calculate the average.

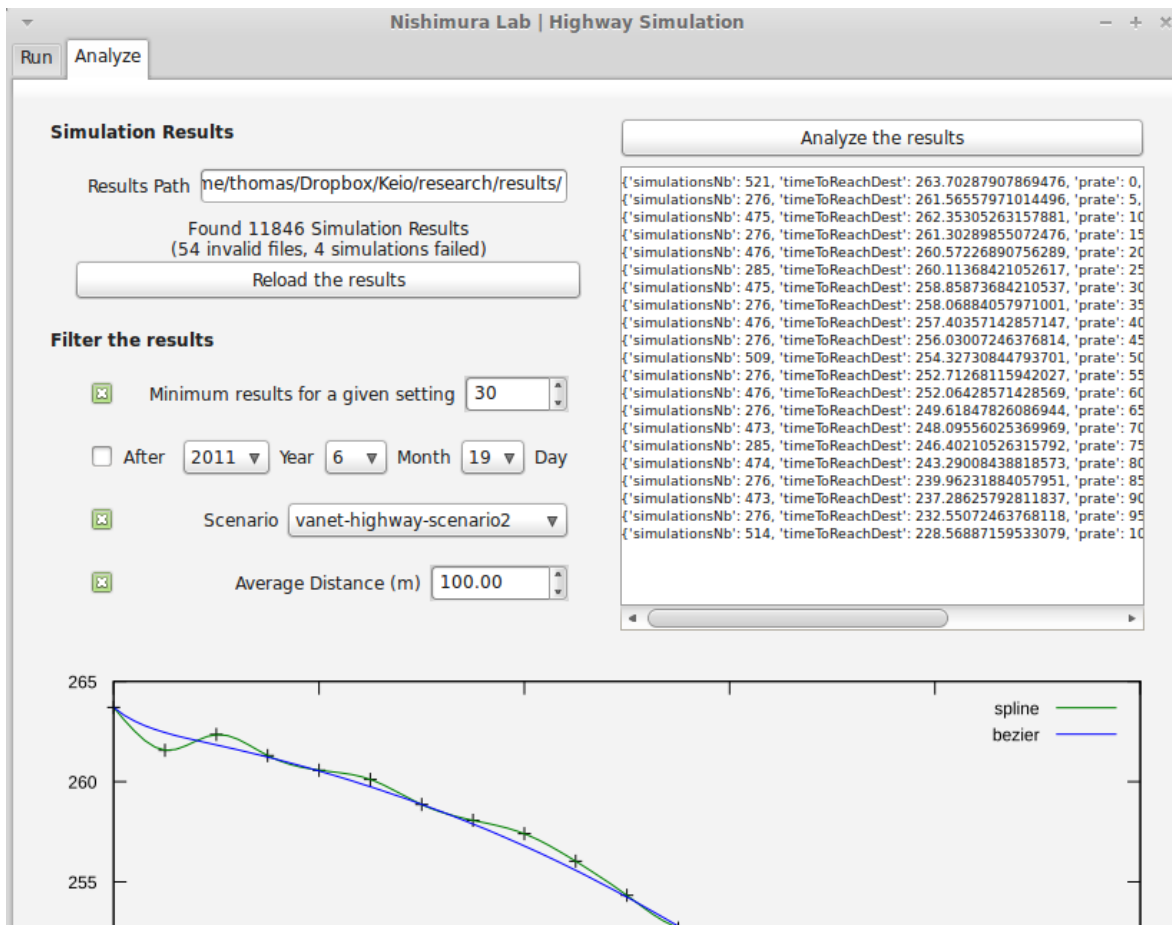


Figure IV.4: Highway Simulation GUI - Analyze

In the screen shot shown in Figure IV.4, on the top right part we can see a textual information about how the plot has been drawn. The first line tells us that for a percentage rate of 0 (no car has the TC4EV application), 521 simulations have been done and the average time to reach the destination is 263.70 seconds.

In addition to the visualization of the graph, an image file (SVG) is created and the raw data used to plot it (in a text file) is also created.

It has been written in Python, and is easily adaptable to one's needs.

The source code of the 3 main classes of the GUI can be found in Appendix C.

We will see the results of the simulation in the next chapter, but before that, let see the simulation environment of the city simulation.

IV.2 City Simulation Environment

As explained previously, the limit of the VANET additions to ns-3 is that it can simulate only a highway. Thus, we need to use a different tool to evaluate the system under a city with intersections.

We chose PreScan to do so. PreScan defines itself as a “software tool designed as a development environment for Advanced Driver Assistance Systems (ADAS) and Intelligent Vehicle (IV) systems. These are systems with sensors that monitor the vehicle’s surroundings and that use the acquired information to take action. Such actions may range from warning the driver of a potentially dangerous situation to actively evading hazards by means of automatic braking or automatic steering.”.

PreScan is scenario-based. It is often used to recreate a real-life scenario where a problem occurred (an accident for example), identify the reasons of the problem and try to solve it by adding additional controls and systems on the vehicles (like sensors or vehicle-to-vehicle communication).

Instead of performing thousands of simulations with random parameters like in the previous case (for the highway simulation), we will only consider a fixed scenario and show how the system works and what are its benefits.

The scenario we chose is shown in Figure IV.5.

The ambulance starts on the left bottom side of the picture and wants to reach the top right part of the picture, following the route highlighted in white.

Other vehicles are on the map (with a red star) and some other vehicles less important will come during the simulation. The ones with a red star on the picture above are important because their actions will influence the ambulance ease of displacement.

We measure the time to reach the destination for the emergency vehicle in both situations:

- without the TC4EV application
- with the TC4EV application



Figure IV.5: City Simulation Scenario Overview

As there is only a few number of vehicles in this simulation, analyzing the performance depending on the penetration percentage of TC4EV is not meaningful. Thus, we will only consider the two cases when all or none of the cars have the TC4EV application.

We will now see the results of both simulations in the next chapter.

V. Results

V.1 Highway Simulation

As introduced previously, we evaluate the system on a highway by simulating the TC4EV application in ns-3, for different number of vehicles per kilometer. More than 30 000 simulations in total have been performed to better appreciate the randomness of some parameters.

a) 10 vehicles per kilometer

First, with an average of 10 vehicles per kilometer, the emergency vehicle needs an average of 263.7 seconds to cover the 10 kilometers, when no car has the TC4EV application. At the opposite extreme, the emergency vehicle needs only 228.6 seconds to cover the 10 kilometers when all the cars have the TC4EV application. This is a non negligible benefit of 35 seconds.

Besides, these values must be compared with the optimal time time reach the destination, which is calculated considering a free road where no cars are on the road so that the emergency vehicle can cover the whole distance at its maximum speed. In such case, the optimal time for the emergency vehicle to cover the 10 kilometers is 225.3 seconds, which means that the system gives a result very close to the optimal time when all the vehicles have the system.

In order to evaluate the benefits of the application when only some vehicles have it, as it is expected to be in a first time, we also simulated the intermediate percentages.

For example when 50 % of the vehicles have the system, the emergency vehicle still needs 254.3 seconds to reach its destination. This is an improvement of only 9 seconds over the case when no car has the system.

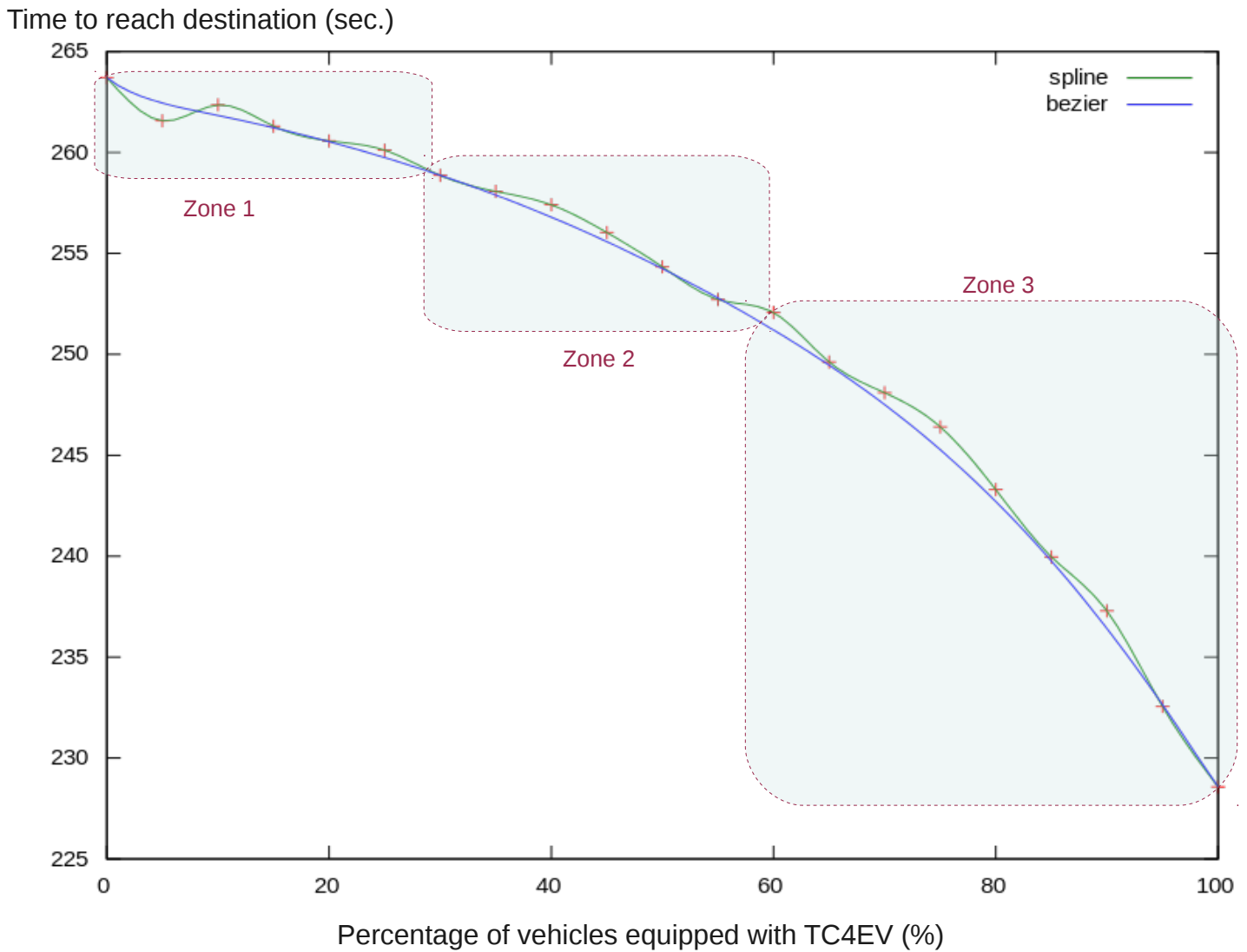


Figure V.1: Highway Simulation | Time to reach destination (10 veh/km)

The detail of the time-to-reach the destination depending on the percentage of vehicles equipped with the TC4EV application is shown in Figure V.1.

The velocity of the emergency vehicle has been recorded for some simulations and the following figure shows an example of the velocity of the emergency vehicle over the time when none of the cars have the TC4EV capability.

We see in Figure V.2 that the emergency vehicle never reaches its maximum desired velocity (165 km/h) and does not have a stable velocity.

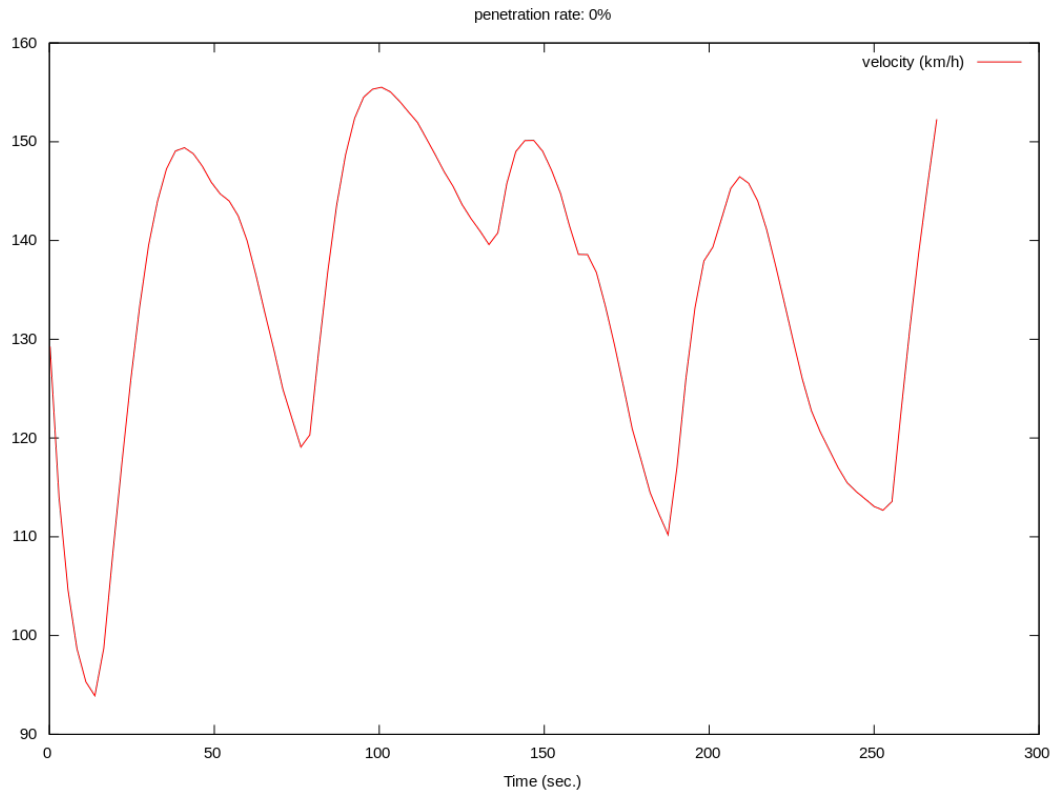


Figure V.2: Velocity of the EV when no car has TC4EV

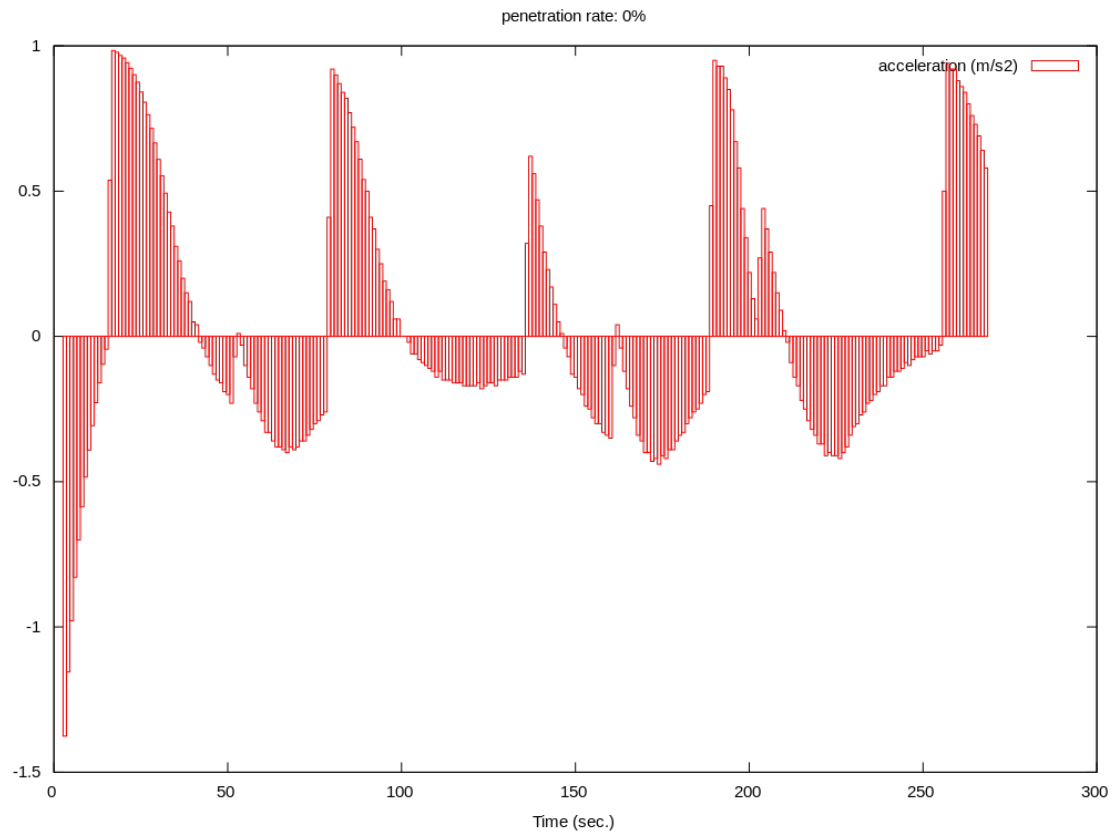


Figure V.3: Acceleration of the EV when no car has TC4EV

Figure V.3 shows the acceleration of the emergency vehicle over the time.

We notice many accelerations and decelerations. The vehicle is always accelerating (positive values) or decelerating (negative values) and almost never keeping the same speed (values close to 0).

At the opposite, Figure V.4 shows an example of the velocity of the emergency vehicle when all the cars have the TC4EV application.

We can see that this time, the emergency vehicle could drive at its desired maximum velocity and does not accelerate nor decelerate often.

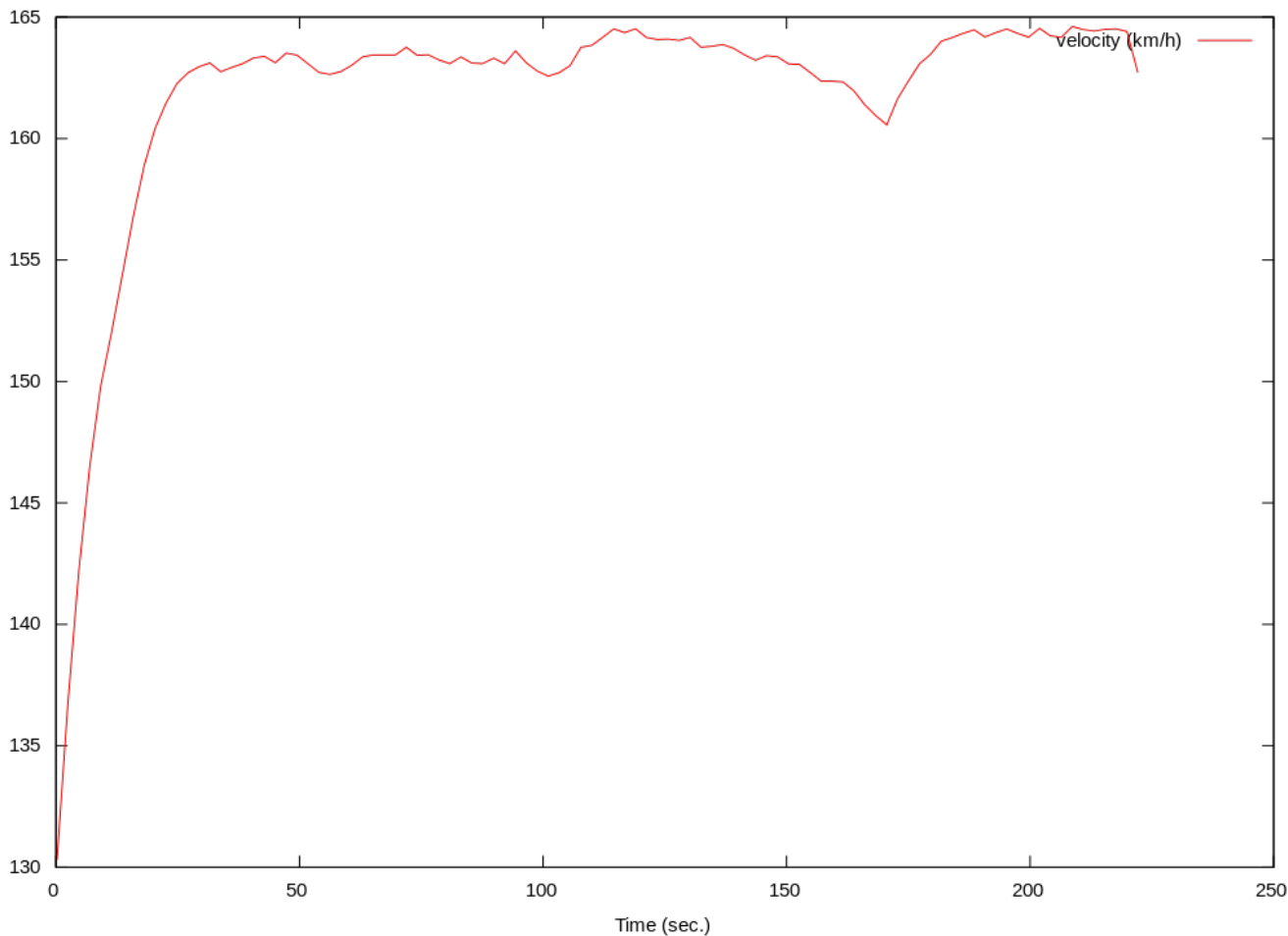


Figure V.4: Velocity of the emergency vehicle when all the cars have TC4EV

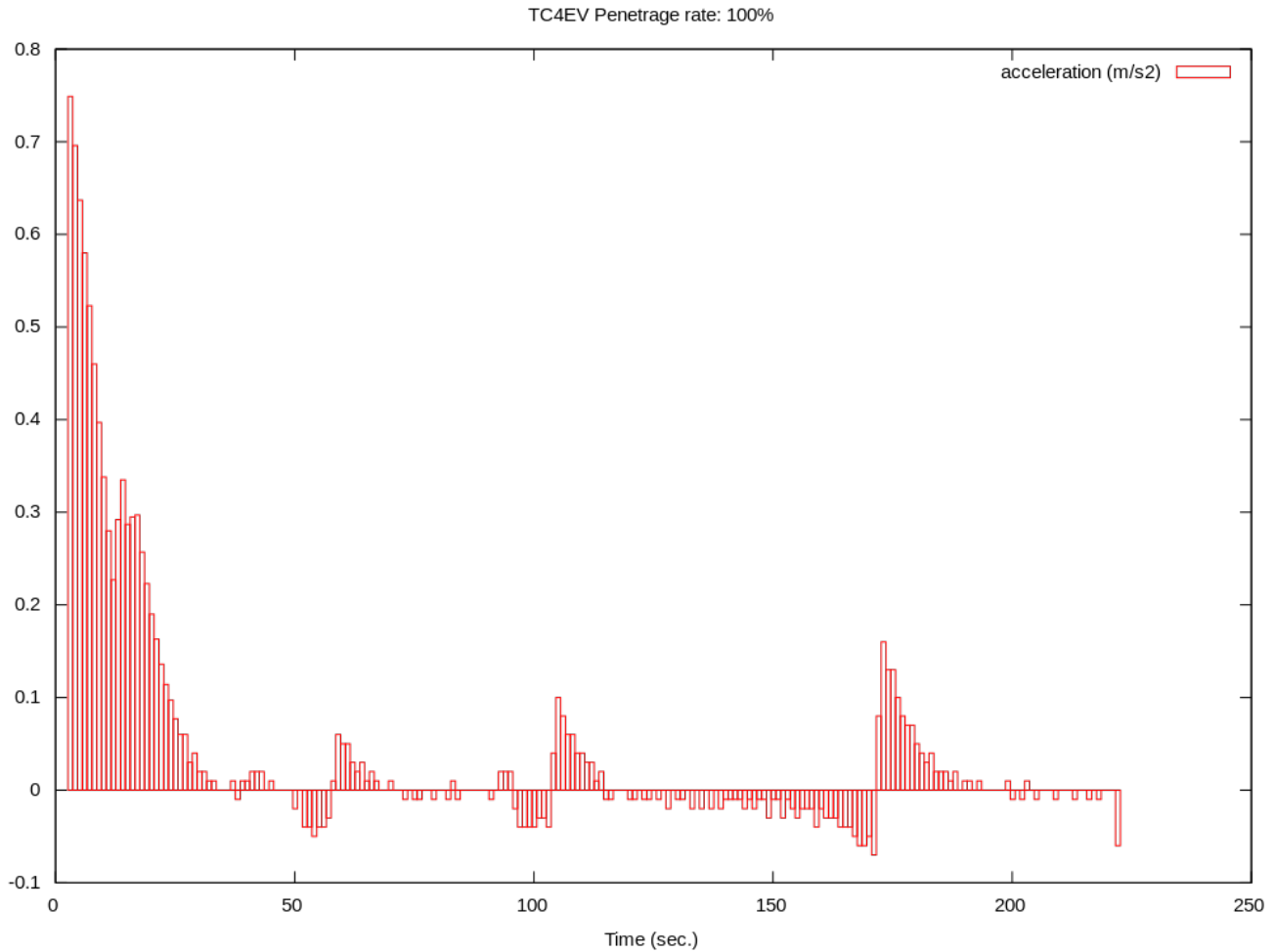


Figure V.5: Acceleration of the emergency vehicle when all the cars have TC4EV

This can be also confirmed by the acceleration of the emergency vehicle shown in Figure V.5.

The vehicle strongly accelerates until reaching its desired maximum speed and keeps it. There are still some low accelerations and decelerations due to the realistic driver model.

Another interesting information is the additional time needed to reach the destination compare to the optimal time (225.3 seconds, when there are no cars in the road).

Figure V.6 shows the additional time to reach the destination compared to the optimal time depending on the TC4EV penetration percentage.

Additional time to reach the destination compared to the optimal time
depending on TC4EV penetration percentage

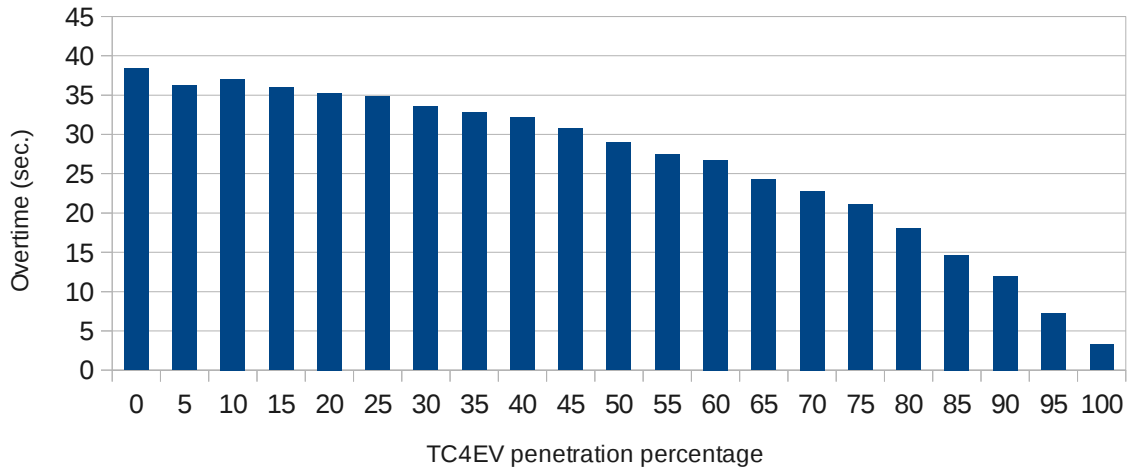


Figure V.6: Additional time to reach destination (10 veh/km)

It goes from 38.4 additional seconds to reach the destination when no cars have the system to only 3.3 seconds when all the cars have the system (close to 0, the optimal value).

We notice 3 different zones that have been highlighted in the Figure 40 page 72:

- Zone 1: from 0% to 30%, small improvements
- Zone 2: from 30% to 60%, the improvements start to be more notable
- Zone 3: from 60% to 100%, exponential improvements

These three zones give a good idea of the benefits of the system over the years.

We can assume that in the near future, when only a few percentage of vehicles are expected to have the TC4EV capability (zone 1, 0%-30%), the benefits of the system will unfortunately not be phenomenal.

Then, when more cars will have the TC4EV capability (zone 2, 30%-60%), the benefits will start to increase faster and be more significant.

Finally, when most of the cars will have the TC4EV applications, the benefits of the

system will quickly increase until reaching its optimal performance when all the cars will have the system.

b) 5 vehicles per kilometer

Similar results have been found with a lower number of vehicles per kilometer. As there are less vehicles, the time to reach the destination when no car has the TC4EV application is less.

As the results are very similar to the previous case (10 veh/km) we do not detail furthermore.

The time to reach the destination depending on the TC4EV penetration rate at 5 vehicles per kilometer is shown in Figure V.7.

We will now introduce the case when there are 20 vehicles per kilometer.

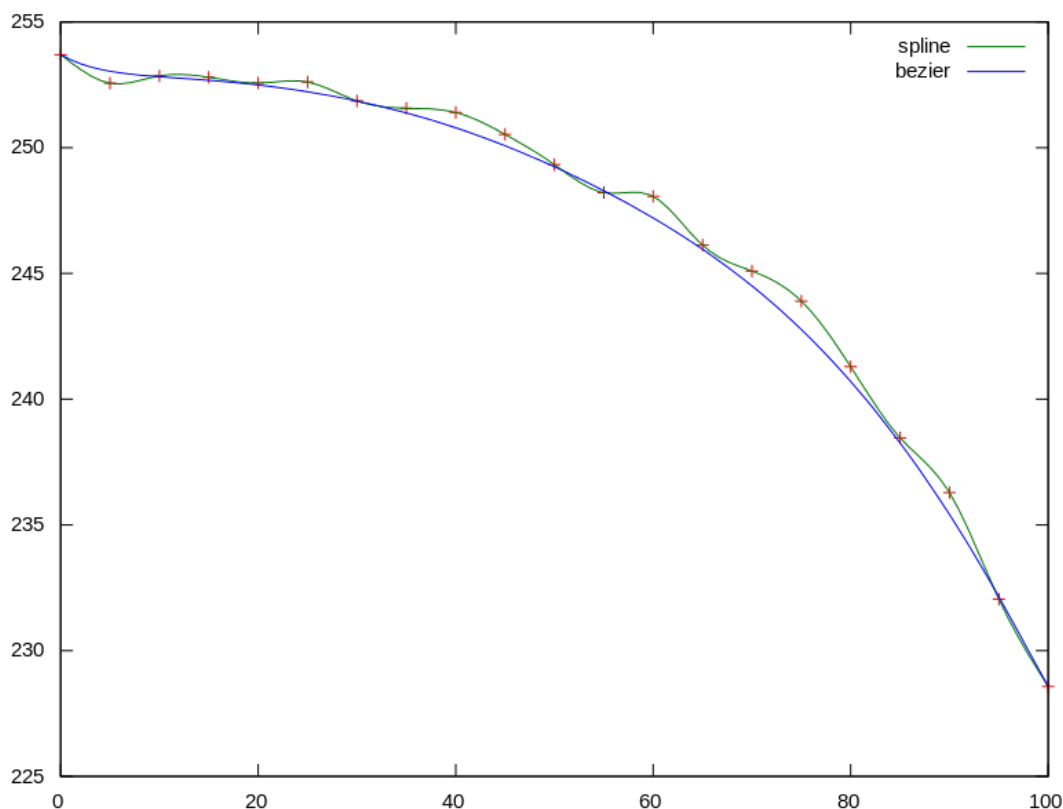


Figure V.7: Time to reach destination (5 veh/km)

c) 20 vehicles per kilometer

When the road is highly congested (20 vehicles per kilometer), the emergency vehicle needs much more time to reach its destination, as shown in Figure V.8.

When none of the cars have the TC4EV capability, the emergency vehicle needs up to 363.7 seconds to reach the destination.

And even when all the vehicles have the TC4EV application, the road is so highly congested anyways that the vehicles often cannot change lane to help the emergency vehicle. In such case, even when all the vehicles have the TC4EV application, the emergency vehicle needs 271.3 seconds to reach its destination.

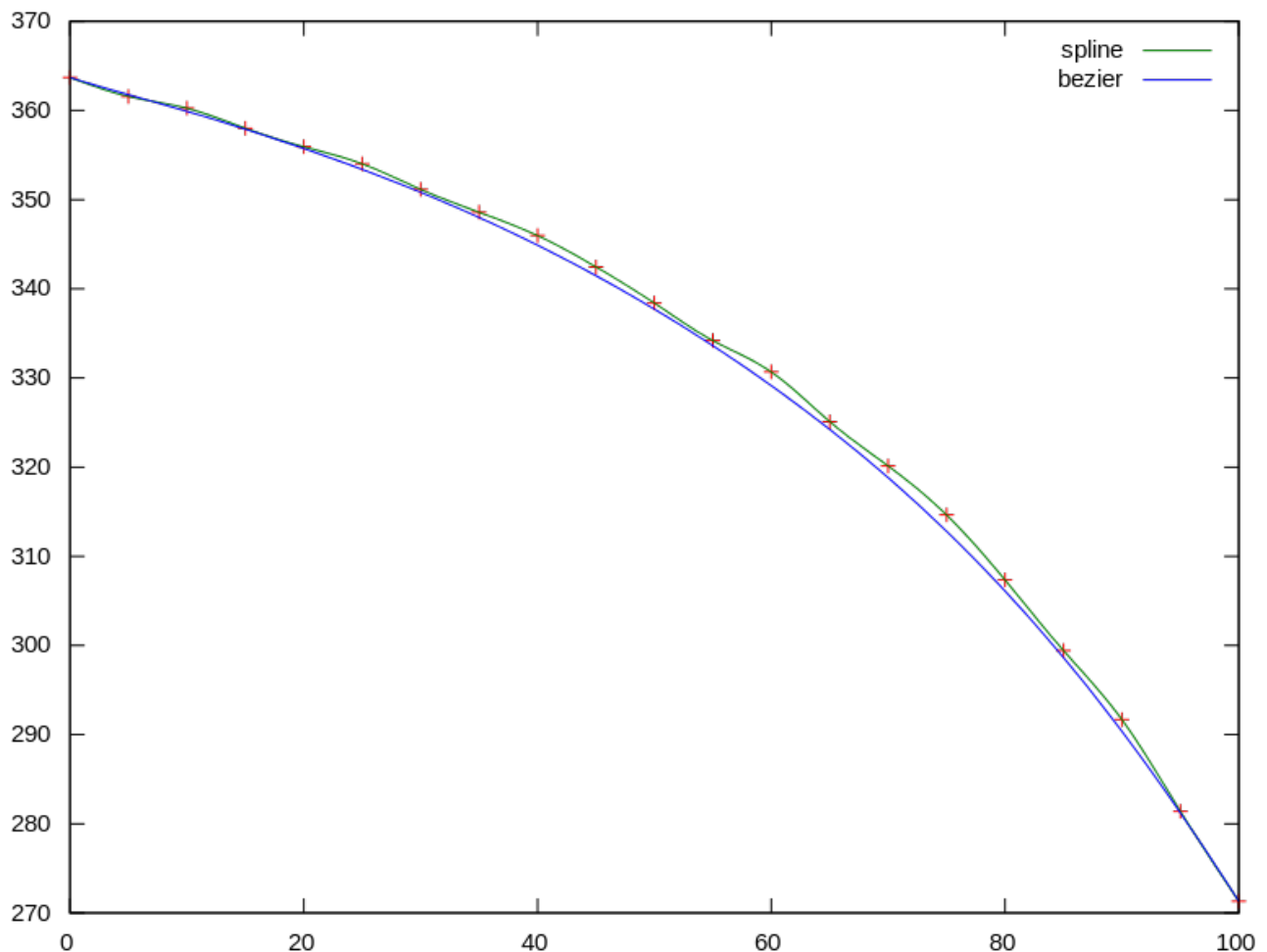


Figure V.8: Highway Simulation | Time to reach destination (20 veh/km)

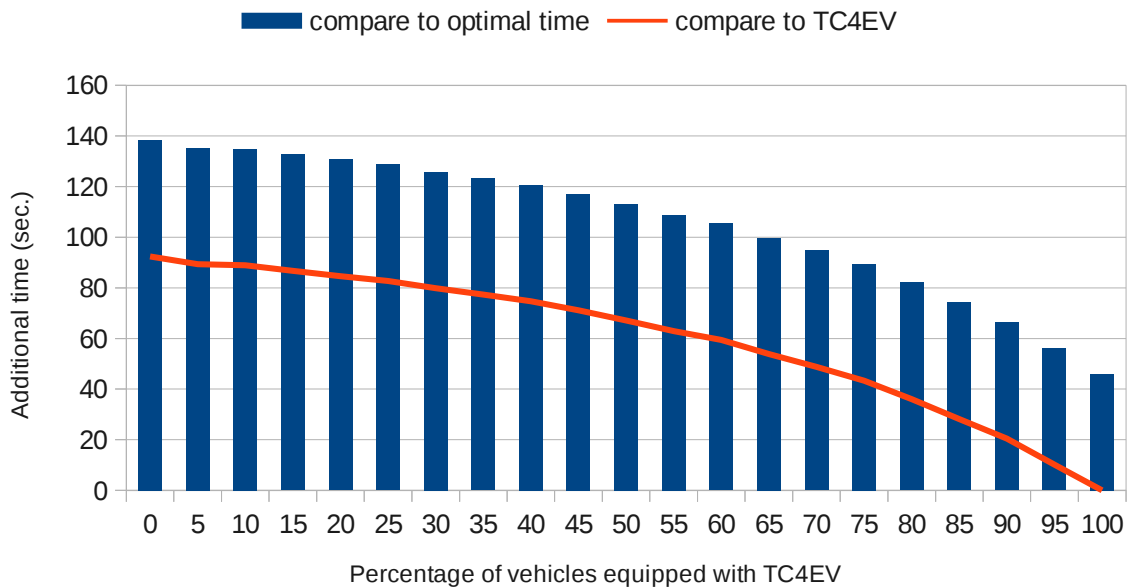


Figure V.9: Additional time to reach destination (20 veh/km)

It is an improvement of 92.4 seconds, a minute and a half, but still 46 seconds slower than the optimal time (without any car). Details of the additional time compared with the optimal time is shown in Figure V.9.

We've learned from these simulations that the more the road is congested, the more time is saved to reach the destination (27 seconds saved at 5 veh/km, 38 at 10 veh/km and 92 at 20 km/h).

It is all the more important that the emergency vehicle drivers waste most of their time when the roads are congested.

We will now present the results of the city evaluation.

V.2 City simulation

The city simulation performed with PreScan compares a given situation in a city with 3 intersections, if the vehicles have the TC4EV system or not.

We will analyze each intersection separately and discuss the benefits of the TC4EV system. Please refer to Figure IV.5 (City Simulation Scenario Overview, page 69) for an

overview of the scenario.

a) Without TC4EV

At the beginning, all the cars want to go to the East direction, as shown in Figure V.10. The black car, arriving first on the priority road, does not see the emergency vehicle and goes straight, as shown in Figure V.11. The blue car also decides to turn right, and, as a result, arriving at the next intersection, the configuration is as shown in Figure V.12.



Figure V.10: 1st intersection, no TC4EV, initial routes

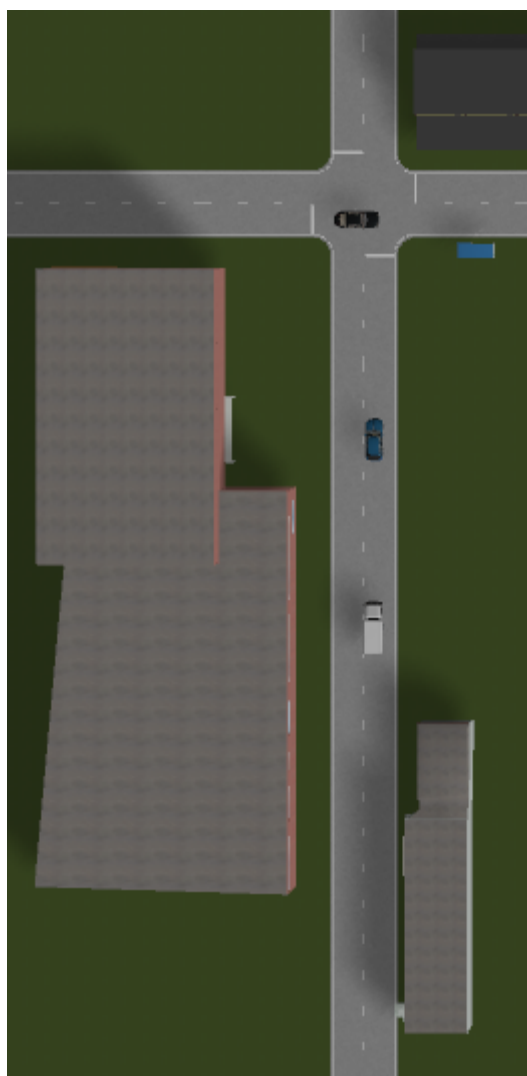


Figure V.11: 1st intersection, no TC4EV, t=2s

The green car in Figure V.12 arrived from the North and will keep going East, as shown in Figure V.13.

The black car will keep going straight while the blue car will gently let the emergency vehicle go first, as shown in Figure V.13.

Arriving at the second intersection, the black car is still in front of the emergency vehicle, and the green car in the bottom right of the previous picture did not see the emergency vehicle so went to the North direction first.

The black car turns right and the emergency vehicle can go left, then straight to reach its destination, as shown in Figure V.14.

However, the green car is limiting the speed of the emergency vehicle that cannot overtake it, as shown in Figure V.15.

As a result, the emergency vehicle took 29.65 seconds to reach its destination.

Let see how we could have improved that time by communicating with the other vehicles through TC4EV to let them know in advance that an emergency vehicle was coming.



Figure V.12: 2nd intersection, no TC4EV, t=7s



Figure V.13: 2nd intersection, no TC4EV, t=10s



Figure V.14: 3rd intersection, no TC4EV, $t=14s$



Figure V.15: Last segment, no TC4EV, $t=23s$

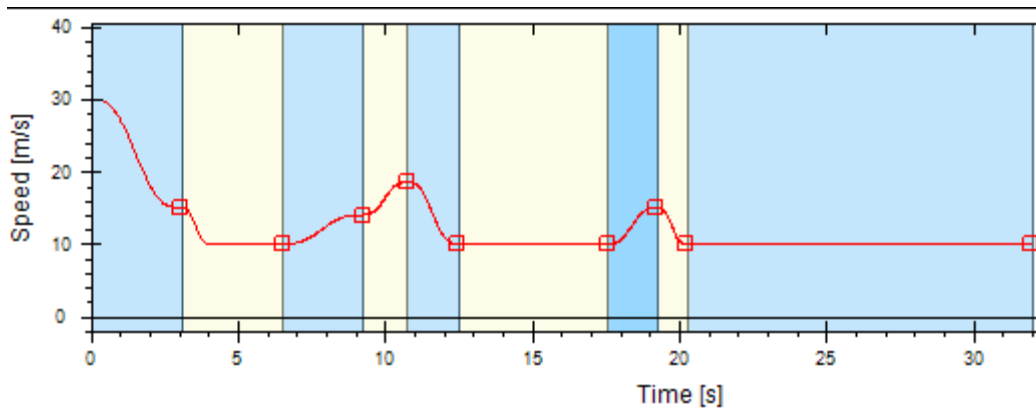


Figure V.16: Velocity of the ambulance over the time (no TC4EV)

Finally, the speed of the ambulance during the simulation is shown in Figure V.16.

We can see from Figure V.16 that the vehicle speed is limited by the vehicles of the preceding vehicles.

b) With TC4EV

With TC4EV, the car navigation system of the surrounding driver receives the information from the emergency vehicle, displays the location and the route the emergency vehicle is going to take and gives a guidance about what the driver should do.

For example at the first intersection, the black car receives a notification that an

emergency vehicle is arriving at the next intersection so that it should stop and let the emergency vehicle go first, as shown in Figure V.17.

The blue car also receives the route information from the ambulance behind it and its car navigation system proposes a new route to the destination that does not cross the ambulance route (by going straight instead of right).

As a result, the emergency vehicle can go first at the intersection, and neither the black car nor the blue one will limit the speed of the emergency vehicle after the intersection. Arriving at the second intersection earlier than without TC4EV, another vehicle that was not important in part (a) because crossing before all the other vehicles, will now receive a notification that an emergency vehicle is arriving and that it should stop at the next intersection to let the emergency vehicle go first, as shown in Figure V.18.



Figure V.17: TC4EV, initial configuration



Figure V.18: 1st intersection, TC4EV, t=4s



Figure V.19: 2nd intersection, TC4EV, t=8s

Thanks to that notification, the green car will slow down and wait at the intersection that the emergency vehicle, going East, goes first.

Finally, another green car, the one that was limiting the speed of the emergency vehicle at the end of the part (a), receives a notification about the coming emergency vehicle and slows down to let the ambulance go first to the North, as shown in Figure V.19.

The ambulance can go first at the third intersection (Figure V.21) and then straight at its maximum speed (Figure V.22).

The ambulance took 15.05 seconds to reach the destination thanks to the TC4EV system, which is almost half of the time it took without the system (29.04).

We also compared this time to the optimal time, when there are no other cars, which is 13.65 seconds.

The 15.05 seconds it took thanks to the TC4EV system are close to the optimal 13.65 seconds.

The speed of the ambulance has been reported in Figure V.20. We can see that thanks to the fact that the other cars receive a warning before the ambulance arrives and leave the way to the ambulance, the ambulance can drive at a higher speed than without the TC4EV application.

In this simulation, we highlighted the benefits of the TC4EV system in cities.

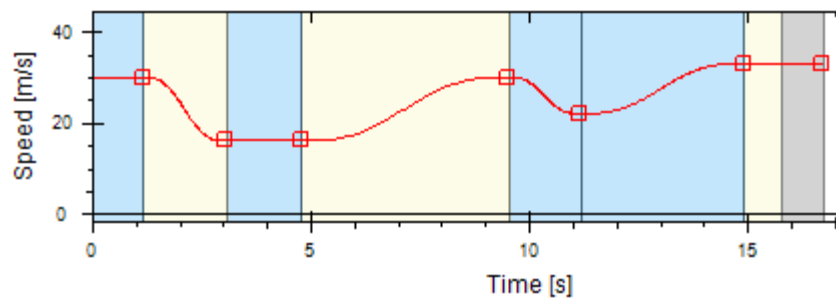


Figure V.20: Velocity of the ambulance over the time (TC4EV)



Figure V.21: 3rd intersection, TC4EV, t=10s



Figure V.22: Last segment, TC4EV, t=13s

VI. Conclusion

In this thesis, we introduced a new ITS application that allows the drivers surrounding an emergency vehicle to receive information about the emergency vehicle (especially its current location and the route it plans to take) and to give directions to the drivers, when possible.

We evaluated the application in two different contexts, the most common ones: a highway and a city. These evaluations showed the application benefits but also its limits. Indeed, the application starts being really efficient only when at least 60% of the vehicles have the system. When less vehicles have the system, there are still some benefits, but small compared to when at least 60% of the vehicles have the system. However, we also introduced a way to implement the system requiring only a mobile phone connected to the internet. Thanks to that, the TC4EV system could be quickly deployed to most of the vehicles, improving its performance.

In the near future, when only some of the cars are expected to have the application, the benefits will not be phenomenal. However, considering that in a distant future all the vehicles will be expected to have the VANET capability, and so on able to use the TC4EV application, we showed that our application can highly reduce the time to reach the destination of the emergency vehicles when most of the vehicles have the application.

We showed that by controlling the traffic to give priority to the emergency vehicle by giving directions to the surrounding drivers, the emergency vehicles can almost reach their optimal traveling time (when no car is on the road except the emergency vehicle). Indeed, by giving directions to the drivers before the emergency vehicles arrive close to them, they can anticipate the arrival of the emergency vehicle and make the right choice in advance to leave the way to the emergency vehicle.

We also found through a questionnaire that most of the people are sometimes confused when there is an emergency vehicle and do not know what to do. 75% of the people who answered to the questionnaire think that an application like TC4EV would help them to make the correct decision. In fact, just the fact of knowing in advance the location and future route of the emergency vehicles is already a great help for the drivers.

VII. Further research

Some improvements of the algorithm could be done in order to give more accurate and detailed information to the driver. In the algorithm presented in this research, we have 5 final states (update display information, change lane, slow down, stop, change route). One can improve the algorithm to satisfy his or her needs.

One can also think about other information to transmit through the application instead of the location, roads IDs, etc... and develop its own decision algorithm to give directions to the drivers.

Another interesting work would be to implement and evaluate multiple ITS applications together. So far, all the researchers design their own application on their own, and evaluate it independently, but it would be interesting to see how it is possible to have different applications running together smoothly (or not) on the vehicle.

Integration of multiple ITS applications that communicate each others is required and would be an interesting topic to investigate for further researches in ITS in general.

References

- [1] U.S. Department of Transportation, *Traffic Safety Facts 2009*, NHTSA's National Center for Statistics and Analysis, September 2010
- [2] World Health Organization (WHO), *Global status report on road safety*, May 2011
- [3] NISSAN SKY PROJECT, <http://www.nissan-global.com/JJP/SAFETY/ITS/SKY/>
- [4] First ACM workshop on Vehicular Ad Hoc Networks (VANET 2004) <http://www.path.berkeley.edu/vanet/>
- [5] Institute of Electrical and Electronics Engineers, *Amendment to Extend the Base IEEE 802.11 WLAN Specification for Wireless Access in Vehicular Environments*, 2010
- [6] WiMaxMAPS® WiMax Deployments <http://www.wimaxmaps.org/>
- [7] Rostam Shirani, Faramarz Hendessi, Mohammad Ali Montazeri and Mohammad Sheikh Zefreh, Absolute Priority for a Vehicle in VANET, *Advances in Computer Science and Engineering, Communications in Computer and Information Science*, 2009, Volume 6, Part 2, 955-959
- [8] Noman Islam, Zubair A. Shaikh and Shahnawaz Talpur, Towards a Grid-based approach to Traffic Routing in VANET, *E-Indus 2008*, 29th April, 2008, Institute of Industrial and Electronics Engineering, Karachi, Pakistan
- [9] Ali Ghazy, Tarik Ozkul, Design and Simulation of an Artificially Intelligent VANET for Solving Traffic Congestion, *MASAUM Journal of Basic and Applied Sciences (MJBAS)* (ISSN 2076-0841) Volume.1 No.2 September 2009, pages 278-283
- [10] Yukihiro Mori, Environmentally-Conscious Adaptive Cruise Control System Design and Analysis of the Effects on Following Vehicles
- [11] Gradinescu, V., Gorgorin, C., Diaconescu, R., Cristea, V., Iftode, L., Adaptive Traffic Lights Using Car-to-car Communication, *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*
- [12] I. Lequerica, P.M. Ruiz and V. Cabrera, Improvement of vehicular communications by using 3G capabilities to disseminate control information, *IEEE Network*, Vol. 24, No. 1, Jan. 2010, Page(s):32-38.
- [13] Matthias Wellens, Burkhard Westphal and Petri Mahonen (2007), Performance Evaluation of IEEE 802.11-based WLANs in Vehicular Scenarios
- [14] Stephan Eichled (2007), Performance Evaluation of the IEEE 802.11p WAVE Communication Standard
- [15] André Costa, Paulo Pedreiras, José Fonseca, João Nuno Matos (2009), Evaluating WiMAX for Vehicular Communication Applications
- [16] Francisco Sánchez, Marc Seguer, Antoni Freixa, Raimond Holze, Klaus Sochaski, Peter Andreas (2001), From Adaptive Cruise Control to Active Safety Systems
- [17] Christoph Mayser, Karl Naab, Walter Kagerer (2006), InterActive Cruise Control - A New Driver Interaction Concept for ACC systems
- [18] IEEE P802.11 - Task Group p
- [19] Valery Naumov, Rainer Baumann and Thomas Gross, *An Evaluation of Inter-Vehicle*

- Ad-Hoc Networks Based on Realistic Vehicular Traces*, MobiHoc '06 (2006)
- [20] Abdelmalik Bachir and Ahderrahim Benslimane, *A Multicast Protocol in Ad hoc Networks Inter-Vehicle Geocast*, IEEE Vehicular Technology Conference 2003 (2003)
- [21] Yuh-Shyan Chen, Yun-Wei Lin and SingLing Lee, *Mobicast protocol in VANETs*, submitted to IEEE International Conference on Communications 2009 (2009)
- [22] M.N. Mariyasagayam, T. Osafune, M. Lenardi, *Enhanced Multi-Hop Vehicular Broadcast (MHVB) for Active Safety Applications*, IEEE (2007).
- [23] Vehicular Ad-hoc Network, http://en.wikipedia.org/wiki/Vehicular_ad-hoc_network, Wikipedia Encyclopedia
- [24] Marc Torrent Moreno, *Inter-Vehicles Communications: Achieving Safety in a Distributed Wireless Environment*, as a thesis for his Doctor degree (2007)
- [25] The innovative Nissan Leaf vehicles, <http://www.nissan.co.uk/vehicles/electric-vehicles/electric-leaf/leaf.html>
- [26] Aamir Hassan, "VANET Simulators Comparison", Technical report, IDE0948, May2009, Master's Thesis in Electrical Engineering
- [27] Hadi Arbabi and Michele C. Weigle, "Highway Mobility and Vehicular Ad-Hoc Networks in ns-3," In Proceedings of the Winter Simulation Conference. Baltimore, MD, December 2010.
- [28] Intelligent Driver Model, http://en.wikipedia.org/wiki/Intelligent_driver_model
- [29] Li-Der Chou, Ching-Chiang Ho, Jui-Ming Chen, *An Early Warning Scheme for Broadcasting Critical Messages using VANET*, International Journal of Ad Hoc and Ubiquitous Computing 2010 - Vol. 6, No.1 pp. 1 - 9
- [30] Francisco J. Martinez, Juan-Carlos Cano, Carlos T. Calafate, Pietro Manzoni, and Jose M. Barrios, *Assessing the feasibility of a VANET driver warning system*, Proceedings of the 4th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks (2009)
- [31] Rawashdeh, Z.Y.; Mahmud, S.M., *Intersection Collision Avoidance System Architecture*, Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE , vol., no., pp.493-494, 10-12 Jan. 2008
- [32] Andrew Chen, Behrooz Khorashadi, Chen-Nee Chuah, Dipak Ghosal, Michael Zhang, *Smoothing vehicular traffic flow using vehicular-based ad hoc networking & computing grid (VGrid)*, 2006 IEEE Intelligent Transportation Systems Conference, Toronto, 2006.

List of Figures

Figure I.1: NISSAN SKY PROJECT Examples.....	11
Figure I.2: Wireless communication between vehicles.....	12
Figure I.3: Researches related to ITS.....	13
Figure I.4: Researches related to V2V.....	13
Figure II.1: Representation of a wireless access point.....	18
Figure II.2: Computer to computer communication through an access point.....	19
Figure II.3: Direct connection between computers using Wifi.....	19
Figure II.4: Schematic Representation of Vehicular Ad-hoc Networks.....	21
Figure II.5: ITS Applications Classification.....	23
Figure II.6: Delays of the main wireless protocols.....	24
Figure II.7: VANET Protocol Stack.....	25
Figure II.8: Example of some ITS Applications.....	25
Figure II.9: Sudden brake warning with sensors.....	26
Figure II.10: Sudden brake warning with wireless communication.....	27
Figure II.11: Accident report example.....	28
Figure II.12: Example of a warning at an intersection.....	28
Figure II.13: Blind spot detection / Lane-change assistant.....	29
Figure II.14: Locality information example.....	30
Figure II.15: Example of e-mail checking with V2I.....	31
Figure II.16: Simplified schema of the variable speed limit system.....	32
Figure II.17: Representation of a simplified adaptable traffic light.....	33
Figure II.18: Representation of a simplified shortest path calculation.....	34
Figure II.19: Obstacle warning example.....	35
Figure II.20: Schematic Representation of Adaptive Cruise Control.....	36
Figure III.1: Vehicle-to-vehicle communication components.....	42
Figure III.2: Schematic representation of the TC4EV application.....	44
Figure III.3: Lack of information if only 2 roads are known.....	45
Figure III.4: Benefit of knowing at least three roads of the EV.....	46
Figure III.5: Example of a revised route to prioritize the EV.....	47
Figure III.6: State-machine representation of the decision algorithm.....	49
Figure III.7: Screenshot of the program to validate the decision algorithm.....	50

Figure III.8: TC4EV example when the car is behind the EV.....	53
Figure III.9: TC4EV example when the EV is behind the car (1).....	54
Figure III.10: TC4EV example when the EV is behind the car on the same lane.....	54
Figure III.11: TC4EV example when the car already passed the intersection.....	55
Figure IV.1: Highway simulation representation.....	62
Figure IV.2: Highway Simulation GUI Components.....	64
Figure IV.3: Highway Simulation GUI - Run.....	66
Figure IV.4: Highway Simulation GUI - Analyze.....	67
Figure IV.5: City Simulation Scenario Overview.....	69
Figure V.1: Highway Simulation Time to reach destination (10 veh/km).....	71
Figure V.2: Velocity of the EV when no car has TC4EV.....	72
Figure V.3: Acceleration of the EV when no car has TC4EV.....	72
Figure V.4: Velocity of the emergency vehicle when all the cars have TC4EV.....	73
Figure V.5: Acceleration of the emergency vehicle when all the cars have TC4EV.....	74
Figure V.6: Additional time to reach destination (10 veh/km).....	75
Figure V.7: Time to reach destination (5 veh/km).....	76
Figure V.8: Highway Simulation Time to reach destination (20 veh/km).....	77
Figure V.9: Additional time to reach destination (20 veh/km).....	78
Figure V.10: 1st intersection, no TC4EV, initial routes.....	79
Figure V.11: 1st intersection, no TC4EV, t=2s.....	79
Figure V.12: 2nd intersection, no TC4EV, t=7s.....	80
Figure V.13: 2nd intersection, no TC4EV, t=10s.....	80
Figure V.14: 3rd intersection, no TC4EV, t=14s.....	81
Figure V.15: Last segment, no TC4EV, t=23s.....	81
Figure V.16: Velocity of the ambulance over the time (no TC4EV).....	81
Figure V.17: TC4EV, initial configuration.....	82
Figure V.18: 1st intersection, TC4EV, t=4s.....	83
Figure V.19: 2nd intersection, TC4EV, t=8s.....	83
Figure V.20: Velocity of the ambulance over the time (TC4EV).....	84
Figure V.21: 3rd intersection, TC4EV, t=10s.....	84
Figure V.22: Last segment, TC4EV, t=13s.....	85

Appendices

APPENDIX A

QUESTIONNAIRE TO THE NORMAL DRIVERS AND EMERGENCY DRIVERS TOWARDS TC4EV

1. What kind of vehicle do you drive?

#	Answer	Response	%
1	Car	122	86%
2	Bus	1	1%
3	Truck	13	9%
4	Motorbike	19	13%
5	Other	8	6%

Statistic	Value
Min Value	1
Max Value	5
Total Responses	142

2. What is your driving experience?

#	Answer	Response	%
1	Less than 1 year	7	5%
2	1 to 3 years	15	11%
3	3 to 5 years	13	9%
4	5 to 10 years	29	20%
5	10 to 20 years	27	19%
6	More than 20 years	51	36%
	Total	142	100%

Statistic	Value
Min Value	1
Max Value	6
Mean	4.46

Variance	2.39
Standard Deviation	1.55
Total Responses	142

3. In a highway, when you notice a priority vehicle, you try to facilitate its moving...

#	Answer	Response	%
1	Always	111	90%
2	Often	12	10%
3	Sometimes	1	1%
4	Almost never	0	0%
5	Never	0	0%
	Total	124	100%

Statistic	Value
Min Value	1
Max Value	3
Mean	1.11
Variance	0.12
Standard Deviation	0.34
Total Responses	124

4. In a highway, when you hear a priority vehicle siren, how long do you usually need to locate it?

#	Answer	Response	%
1	Less than 2 seconds	11	9%
2	2 to 3 seconds	43	35%
3	3 to 5 seconds	53	43%
4	More than 5 seconds	17	14%
	Total	124	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	2.61
Variance	0.69

Standard Deviation	0.83
Total Responses	124

5. In a highway, how fast do you consider your reaction after locating an emergency vehicle?

#	Answer	Response	%
1	Fast (right after locating it)	16	13%
2	Normal (time to analyze the situation)	107	86%
3	Slow	0	0%
4	Very slow	1	1%
5	No reaction	0	0%
	Total	124	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	1.89
Variance	0.15
Standard Deviation	0.39
Total Responses	124

6. When you detect an emergency vehicle behind you in a highway and decide to facilitate the moving of the emergency vehicle, you have to make a decision (change lane, stay here...).

#	Answer	Response	%
1	It is easy (I always know what to do to facilitate the moving of the emergency vehicle)	73	59%
2	It is not so easy and I sometimes do not know what I should do	46	37%
3	I often do not know what I should do	5	4%
4	I do not worry because I do not do anything anyway	0	0%
	Total	124	100%

Statistic	Value
Min Value	1
Max Value	3

Mean	1.45
Variance	0.33
Standard Deviation	0.58
Total Responses	124

7. If you have any comment about emergency vehicles in a highway, please write it in the following box (optional):

Text Response

サイレンの音が聞こえてから実際に車の後ろに来るまでの間が短い。また、路肩を走ってくれと助かる。

あまり遭遇しない。

自分の車の防音性が高く、緊急車両のサイレンが聞こえない時が度々ある。音を大きめにして音楽を聴いていると余計にサイレンに気づかない。

Although I have answered this way, I have never come across an emergency vehicle on the highway in the past.

渋滞しているときの対応が難しい。たとえば、2車線の首都高で右車線を走行中に、右車線後方から緊急車両が来たとき、左車線に移動しようと思っても左車線が混雑しているためすぐに車線変更ができない。

音が小さいため、近くまでこないとあまり気づかない。

Les réponses sont très différentes en cas de bouchons où la visibilité devient mauvaise : il est alors parfois plus difficile de voir d'où vient le véhicule prioritaire et par où il compte passer.

I answered "3 to 5 seconds" but really difficult to estimate.

運転経験が短く、あまり遭遇したことがございません。

あまり緊急車両に遭遇した事ない。逆に高速ではオービスと覆面に気をつけて運転してる。

Dur d'entendre la sirène.

あまり見たことがないので、わからない。 I rarely see the emergency car in a highway, so I cannot say anything....

そもそも高速道路で緊急車両と遭遇することが稀です。またサイレンに気付いた場合の多くは反対車線の対向車であり何も対処する必要がありません。自分の車の後方から緊急車両が来た場合は隣の車線の安全性などを確認したうえで車線変更します。

パトカーがサイレンを鳴らして追い越してゆくが、何が緊急なのかわからない(知らされない)ことが多い

あまり遭遇したことない

交通状況(込み合っていてすぐに車線変更できない等々)で、緊急車両に適切なサポートができない時があります。また、すぐに車線変更することが事故を誘発することもあり、この研究はとても有意義なものと思います。 成果を出されることを願っております。

一般道よりもサイレンが聞こえにくい(走行音などの騒音が大きい)ため、運転者は赤色灯(パトランプ)をバックミラー(サイドミラー)で認識して、はじめて緊急車であることがわかると思います。

高速走行時のエンジン音でサイレンが聞こえない時もある。サイレンより先に緊急車両を見つけることも多いです。しかし目視が確認して適切に対処する為の十分な時間はあります。

遭遇したことがないので、想像で書きました。

The difficulty of facilitating varies sometimes according to the location. For example, Facilitating for emergency vehicles within the USA highways was much easier than Saudi Arabia. In USA, Speed limit and regulations are more stricter. Also, the size of the highway is wider than the size of highways within Saudi Arabia.

サイレンが聞こえない

消防隊、救急隊は追い越し車線を走行する一般車より、走行スピードが遅いので、あいまいな走行になる時がある。

緊急車両の種類と仕事内容【役割】が一般的にしられていないように感じる。

あまり高速道路で緊急車両に遭遇したことがない

Statistic	Value
Total Responses	24

8. In a city, when you notice a priority vehicle, you try to facilitate its moving...

#	Answer	Response	%
1	Always	110	95%
2	Often	6	5%
3	Sometimes	0	0%
4	Almost never	0	0%
5	Never	0	0%
	Total	116	100%

Statistic	Value
Min Value	1
Max Value	2
Mean	1.05
Variance	0.05
Standard Deviation	0.22
Total Responses	116

9. In a city, when you hear a priority vehicle siren, how long do you usually need to locate it?

#	Answer	Response	%
1	Less than 2 seconds (very quick)	11	9%
2	2 to 3 seconds (quick)	46	40%
3	3 to 5 seconds	37	32%
4	More than 5 seconds	22	19%
	Total	116	100%

Statistic	Value
-----------	-------

Min Value	1
Max Value	4
Mean	2.60
Variance	0.82
Standard Deviation	0.90
Total Responses	116

10. In a city, how fast do you consider your reaction after locating an emergency vehicle?

#	Answer	Response	%
1	Fast (right after locating it)	20	17%
2	Normal (after taking time to analyze the situation)	93	80%
3	Slow (I don't really pay attention to the priority vehicles)	1	1%
4	Very Slow (only if I am really blocking a priority vehicle just behind me)	2	2%
5	No Reaction	0	0%
	Total	116	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	1.87
Variance	0.24
Standard Deviation	0.49
Total Responses	116

11. When you detect an emergency vehicle behind you in a city and decide to facilitate the moving of the emergency vehicle, you have to make a decision (change lane, stay here...).

#	Answer	Response	%
1	It is easy (I always know what to do to facilitate the moving of the emergency vehicle)	67	58%
2	It is not so easy and I sometimes do not know what I should do	47	41%
3	I often do not know what I should do	2	2%
4	I do not worry because I do not do anything anyway	0	0%

Total		116	100%
-------	--	-----	------

Statistic	Value
Min Value	1
Max Value	3
Mean	1.44
Variance	0.28
Standard Deviation	0.53
Total Responses	116

12. If you have any comment about emergency vehicles in a city, please write it in the following box (optional):

Text Response

街の道路、2車線の場合だと、避ける方向が歩道の方だったり、対向車線方向へよって停止すればいいので、比較的簡単にできる。ただ、周りの2輪を考えないといけないからそこは気をを使う。

都内だと対応が難しい

渋滞している道路で後方から緊急車両が来たとき、道を譲りたいが退避するスペースがなくどうにもならない時がある。

停止するか、道をあげながら走行するか、ほかの車の動きをみながらどちらにすべきかを迷うことが多い。緊急車両のサイレンの音が小さいので、音が聞こえた時にはすぐそばに緊急車両が来ていることがある。

車内で音楽を聞いている場合は、わからないことがあります。

こちらは、気づき次第、自分の車がどの様にすればいいのかは直感的にはわかる。しかし、最近車で音楽を聞く事が多いので、認識まで時間がかかる。

En ville c'est plus difficile de localiser les ambulances.

大抵横によけるようにしている。 always go to side of the way

サイレンに気付いてから車両を認識するまでは、渋滞の場合などはより多くの時間がかかると思います。退避の場合の多くは左車線や左側の路肩に移動しますが、道路上の車両が多くて右側車線の場合は右側に寄せて退避する場合もあります。

自分がどのように行動すれば良いかは、他車の行動に依存する場合があるため、判断が難しいときもある。

都内だと、緊急車両自体そこまで早いスピードで迫ってこない。

サイレン音は早い段階で認知できるが、車両の位置の特定は難しい

こちらも、遭遇したことがないので、想像で書きました。

通常は左に寄り、右側を緊急車両用に空けるものだが、ケースによってはその逆をした方が緊急車両が通りやすいと思うことがある。そのような時どちらに寄るべきか判断に迷うことが時々ある。

Statistic	Value
Total Responses	14

13. How often do you see emergency vehicles (ambulances, police cars, fire trucks...)?

#	Answer	Response	%
1	Everyday	12	10%

2	Almost everyday		38	33%
3	Once a week		36	31%
4	Once a month		18	16%
5	Almost never		12	10%
	Total		116	100%

Statistic	Value
Min Value	1
Max Value	5
Mean	2.83
Variance	1.29
Standard Deviation	1.14
Total Responses	116

14. When there is an emergency vehicle siren and/or light, do you loose concentration on your driving?

#	Answer	Response	%
1	Yes, I always pay attention to the emergency vehicle at the expense of my own driving	3	3%
2	Yes, I sometimes pay too much attention to the emergency vehicle	26	22%
3	Yes but I pay attention to both my driving and the emergency vehicle	81	70%
4	No, I always focus first on my own driving	6	5%
	Total	116	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	2.78
Variance	0.33
Standard Deviation	0.58
Total Responses	116

15. Have you ever had an accident or a problem because you lost concentration because of an emergency vehicle siren or lights?

#	Answer	Response	%
---	--------	----------	---

1	Yes, more than once		0	0%
2	Yes, one time		0	0%
3	No, never		116	100%
	Total		116	100%

Statistic	Value
Min Value	3
Max Value	3
Mean	3.00
Variance	0.00
Standard Deviation	0.00
Total Responses	116

16. Do you think this notification can help you? *EV = Emergency Vehicle

#	Question	Yes	N/A	No	Responses	Mean
1	I will not lose time looking for the EV	85	23	6	114	1.31
2	I will not lose concentration looking for the EV	57	45	12	114	1.61
3	I will not lose time finding out what to do to facilitate the way to the EV	72	31	11	114	1.46
4	I will not lose concentration finding out what to do to facilitate the way to the EV	59	45	10	114	1.57
5	It will help me to make the right decision to help the emergency vehicle	90	17	7	114	1.27

Statistic	I will not lose time looking for the EV	I will not lose concentration looking for the EV	I will not lose time finding out what to do to facilitate the way to the EV	I will not lose concentration finding out what to do to facilitate the way to the EV	It will help me to make the right decision to help the emergency vehicle
Min Value	1	1	1	1	1
Max Value	3	3	3	3	3
Mean	1.31	1.61	1.46	1.57	1.27
Variance	0.32	0.45	0.45	0.42	0.32
Standard Deviation	0.57	0.67	0.67	0.65	0.57
Total Responses	114	114	114	114	114

17. Are you eager to follow the directions given by such car navigation system (in the case it makes sense and is feasible)?

#	Answer	Response	%
---	--------	----------	---

1	Yes		92	81%
2	No		22	19%
	Total		114	100%

Statistic	Value
Min Value	1
Max Value	2
Mean	1.19
Variance	0.16
Standard Deviation	0.40
Total Responses	114


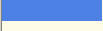


18. If your car vehicle has this kind of notification system, how will you react to this notification?

#	Answer	Response	%
1	I just follow blindly the car navigation directions	2	2%
2	If it is feasible, I just follow the directions and do not waste time analyzing the complete situation	28	25%
3	I still double check by myself the situation (as if I did not have received any notification) to really make sure this is the right thing to do	78	68%
4	I ignore the car navigation suggestion and make the decision by myself	6	5%
	Total	114	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	2.77
Variance	0.32
Standard Deviation	0.56
Total Responses	114

19. Please rate your general enthusiasm to this notification system (from 5-Great to 1-Useless)

#	Answer	Response	%
1	5 - Great (I would love to have it!)	9	8%

2	4 - Interested		67	59%
3	3 - No more and no less		24	21%
4	2 - Not really interested		12	11%
5	1 - Useless (I do not want it)		2	2%
	Total		114	100%

Statistic	Value
Min Value	1
Max Value	5
Mean	2.39
Variance	0.72
Standard Deviation	0.85
Total Responses	114

20. Have you ever driven an emergency vehicle?

#	Answer	Response	%
1	Yes, an ambulance	23	20%
2	Yes, a police car	1	1%
3	Yes, a fire-truck	34	30%
4	Yes, another emergency vehicle	2	2%
5	No	63	55%

Statistic	Value
Min Value	1
Max Value	5
Total Responses	114

21. Generally, do other drivers react positively (helping you) to your presence?

#	Answer	Response	%
1	Always	1	2%
2	Often	40	78%
3	Sometimes	8	16%
4	Almost never	2	4%

5	Never		0	0%
Total			51	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	2.22
Variance	0.29
Standard Deviation	0.54
Total Responses	51

22. When driving in a highway, do other drivers facilitate you the way?

#	Answer	Response	%	
1	Yes, all of them		1	2%
2	Yes, most of them		30	59%
3	Yes, some of them		17	33%
4	No, almost none of them		3	6%
5	No, none of them		0	0%
Total		51	100%	

Statistic	Value
Min Value	1
Max Value	4
Mean	2.43
Variance	0.41
Standard Deviation	0.64
Total Responses	51

23. When people try to facilitate you the way, do they manage to do it?

#	Answer	Response	%	
1	They always make the right decision to help me		1	2%
2	They sometimes make the wrong decision so it does not help me		33	65%
3	They often make the wrong decision and slow me down		17	33%
4	They never make the right decision and always slow me down!		0	0%

Total	51	100%
-------	----	------

Statistic	Value
Min Value	1
Max Value	3
Mean	2.31
Variance	0.26
Standard Deviation	0.51
Total Responses	51

24. In a highway, what is the average distance of the other cars when they make a move (to help you)? When you are...

#	Answer	Response	%
1	10 meters behind (when I arrive very close)	3	6%
2	10 to 30 meters behind (close)	15	29%
3	30 to 60 meters behind (average distance between two cars on a highway)	28	55%
4	60 to 100 meters behind	5	10%
5	100 to 150 meters behind	0	0%
6	More than 150 meters behind (fast reaction when I am still far away)	0	0%
Total		51	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	2.69
Variance	0.54
Standard Deviation	0.73
Total Responses	51

25. Do you think that if people received notifications and directions on their car navigation system, it would help you to reach your destination faster?

#	Answer	Response	%
1	Yes, definitely!	2	4%
2	Yes, probably	21	41%
3	Yes, maybe	19	37%

4	No, not really		9	18%
5	No		0	0%
Total			51	100%

Statistic	Value
Min Value	1
Max Value	4
Mean	2.69
Variance	0.66
Standard Deviation	0.81
Total Responses	51

26. As an emergency vehicle driver, please rate your general enthusiasm to this notification system (from 5-Great to 1-Useless)

#	Answer	Response	%
1	5 - Great! (I would love to have it!)	2	4%
2	4 - Interested	33	65%
3	3 - No more and no less	12	24%
4	2 - Not really interested	4	8%
5	1 - Useless (I do not want it)	0	0%
Total			51

Statistic	Value
Min Value	1
Max Value	4
Mean	2.35
Variance	0.47
Standard Deviation	0.69
Total Responses	51

APPENDIX B

ALGORITHM VALIDATION SOURCE CODE

```
__author__ = "Thomas Dall'Agnese"

from PySide.QtGui import *
from PySide.QtCore import *

from helpers import *
from math import sqrt

# distance between two points:
def dist(p1,p2):
    x1 = p1.x()
    y1 = p1.y()
    x2 = p2.x()
    y2 = p2.y()
    return sqrt( (x2 - x1)**2 + (y2 - y1)**2 )

def lineDist(l):
    return dist(l.p1(),l.p2())

def atIntersection(seg):
    if seg=='Q' or seg=='R' or seg=='S' or seg=='T':
        return True
    else:
        return False

def sameWay(seg1, seg2):
    if seg1==seg2\
    or (seg1=='A' and seg2=='B')\
    or (seg1=='B' and seg2=='A')\
    or (seg1=='C' and seg2=='D')\
    or (seg1=='D' and seg2=='C')\
    or (seg1=='E' and seg2=='F')\
    or (seg1=='F' and seg2=='E')\
    or (seg1=='G' and seg2=='H')\
    or (seg1=='H' and seg2=='G')\
    or (seg1=='I' and seg2=='J')\
    or (seg1=='J' and seg2=='I')\
    or (seg1=='K' and seg2=='L')\
    or (seg1=='L' and seg2=='K')\
    or (seg1=='M' and seg2=='N')\
```

```

    or (seg1=='N' and seg2=='M')\
    or (seg1=='O' and seg2=='P')\
    or (seg1=='P' and seg2=='O'):
        return True
else:
    return False

def routesIntersect(path1, path2):
    for seg1 in path1:
        for seg2 in path2:
            if seg1 == seg2:
                return True
    return False

def redirect(carSeg,dst):
    # if the cars are in the GH segment
    if carSeg=='G' or carSeg=='H':
        if dst == 'W':
            return 'N'
        elif dst != 'E':
            return 'E'
        else:
            return 'N'
    elif carSeg=='A' or carSeg=='B':
        if dst=='E':
            return 'S'
        elif dst != 'W':
            return 'W'
        else:
            return 'S'
    elif carSeg=='K' or carSeg=='L':
        if dst == 'N':
            return 'E'
        elif dst != 'S':
            return 'S'
        else:
            return 'E'
    elif carSeg=='M' or carSeg=='N':
        if dst == 'S':
            return 'W'
        elif dst == 'W':
            return 'N'
        else:
            return 'W'

```

```

import sys

app = QApplication(sys.argv)
app.setApplicationName("Algorithm Validation")

dashPen = QPen(Qt.white)
dashPen.setStyle(Qt.DashLine)
dashPenRed = QPen(Qt.red)
dashPenRed.setStyle(Qt.DashLine)

class AlgorithmValidationWidget(QWidget):
    message = Signal(str)
    def __init__(self):
        QWidget.__init__(self)
        self.resize(300,300)
        self.setAttribute(Qt.WA_PaintOutsidePaintEvent)
    def paintEvent(self, paintEvent):
        painter = QPainter(self)
        painter.setPen(Qt.black)
        painter.setFont(QFont("Arial", 8))

        # white background
        # background = QRectF(0,0,300,300)
        # painter.drawRect(background)
        # painter.fillRect(background, Qt.white)

        # Roads
        horizontalRoad = QRectF(0,120,300,60)
        painter.fillRect(horizontalRoad, Qt.gray)
        verticalRoad = QRectF(120,0,60,300)
        painter.fillRect(verticalRoad, Qt.gray)
        painter.setPen(Qt.white)
        painter.drawLine(QLineF(0, 150, 120, 150))
        painter.drawLine(QLineF(180, 150, 300, 150))
        painter.drawLine(QLineF(150, 0, 150, 120))
        painter.drawLine(QLineF(150, 180, 150, 300))
        painter.setPen(dashPen)
        painter.drawLine(0, 135, 120, 135)
        painter.drawLine(0, 165, 120, 165)
        painter.drawLine(180, 135, 300, 135)
        painter.drawLine(180, 165, 300, 165)
        painter.drawLine(135, 0, 135, 120)

```

```

painter.drawLine(165, 0, 165, 120)
painter.drawLine(135, 180, 135, 300)
painter.drawLine(165, 180, 165, 300)

self.xSeg = {'A':125,'B':140,'C':155,'D':170,
            'E':125,'F':140,'G':155,'H':170,
            'I':5,'J':5,'K':5,'L':5,
            'M':290,'N':290,'O':290,'P':290,
            'Q':132,'R':162,'S':132,'T':162}
self.ySeg = {'A':10,'B':10,'C':10,'D':10,
            'E':295,'F':295,'G':295,'H':295,
            'I':130,'J':145,'K':160,'L':175,
            'M':130,'N':145,'O':160,'P':175,
            'Q':140,'R':140,'S':168,'T':168}

# Segment names
painter.setPen(Qt.darkGray)
painter.drawText(125,10,'A')
painter.drawText(140,10,'B')
painter.drawText(155,10,'C')
painter.drawText(170,10,'D')
painter.drawText(125,295,'E')
painter.drawText(140,295,'F')
painter.drawText(155,295,'G')
painter.drawText(170,295,'H')
painter.drawText(5,130,'I')
painter.drawText(5,145,'J')
painter.drawText(5,160,'K')
painter.drawText(5,175,'L')
painter.drawText(290,130,'M')
painter.drawText(290,145,'N')
painter.drawText(290,160,'O')
painter.drawText(290,175,'P')
painter.drawText(132, 140, 'Q')
painter.drawText(162, 140, 'R')
painter.drawText(132, 168, 'S')
painter.drawText(162, 168, 'T')

def placeCars(self, carSegment, carSegmentDiff, carDest, ambuSegment, ambuSegmentDiff,
              ambuDest, multipleLanes=True):
    #self.paintEvent('car')
    self.paintEvent('')
    # DEFAULT MESSAGE
    naviMessage = 'DISPLAY INFORMATION ONLY (1)'

```

```

if carSegment==ambuSegment and (carSegmentDiff==ambuSegmentDiff or carSegment=='Q'\
    or carSegment=='R' or carSegment=='S' or carSegment=='T'):
    QMessageBox.critical(self, 'Error',
        'The car and the ambulance are at the same position.\
\nUse DIFF to put them on the same segment, different position.')
    return
painter = QPainter(self)
painter.setPen(Qt.black)
carX = 0
carY = 0

# POSITION OF THE POINTS
xPos = {'A':125,'B':140,'C':155,'D':170,
        'E':125,'F':140,'G':155,'H':170,
        'I':60,'J':60,'K':60,'L':60,
        'M':240,'N':240,'O':240,'P':240,
        'Q':132,'R':162,'S':132,'T':162}
yPos = {'A':60,'B':60,'C':60,'D':60,
        'E':240,'F':240,'G':240,'H':240,
        'I':125,'J':140,'K':155,'L':165,
        'M':125,'N':140,'O':155,'P':165,
        'Q':130,'R':130,'S':160,'T':160}

# CALCULATE THE CAR POSITION
carX = xPos[carSegment]
carY = yPos[carSegment]
if carSegment=='A' or carSegment=='B' or carSegment=='C' or carSegment=='D'\
    or carSegment=='E' or carSegment=='F' or carSegment=='G' or carSegment=='H':
    carY += float(carSegmentDiff)*20
elif carSegment!='Q' and carSegment!='R' and carSegment!='S' and carSegment!='T':
    carX += float(carSegmentDiff)*20

# CALCULATE THE AMBULANCE POSITION
ambuX = xPos[ambuSegment]
ambuY = yPos[ambuSegment]
ifambuSegment=='A' orambuSegment=='B' orambuSegment=='C' orambuSegment=='D'\
    orambuSegment=='E' orambuSegment=='F' orambuSegment=='G' orambuSegment=='H':
   ambuY += float(ambuSegmentDiff)*20
elifambuSegment!='Q'andambuSegment!='R' andambuSegment!='S' andambuSegment!='T':
   ambuX += float(ambuSegmentDiff)*20

# POSSIBLE DIRECTIONS
direction = dict()

```

```

direction['W'] = [ ['A', 'Q', 'I'], ['B','Q','J'],
                  ['M', 'R', 'Q', 'I'], ['N','R','Q','J'],
                  ['G', 'T', 'R', 'Q', 'J'], ['H', 'T', 'R', 'Q', 'I'],
                  ['I'], ['J'],
                  ['Q', 'I'], ['R', 'Q', 'I'], ['T', 'R', 'Q', 'I']]
direction['E'] = [ ['B', 'Q', 'S', 'T', 'O'], ['A', 'Q', 'S', 'T', 'P'],
                  ['K', 'S', 'T', 'O'], ['L', 'S', 'T', 'P'],
                  ['G', 'T', 'O'], ['H', 'T', 'P'],
                  ['T', 'P'], ['S', 'T', 'P'], ['Q', 'S', 'T', 'P'],
                  ['E'], ['F']]
direction['N'] = [ ['G', 'T','R', 'C'], ['H', 'T', 'R', 'D'],
                  ['M', 'R', 'D'], ['N', 'R', 'C'],
                  ['K', 'S', 'T', 'R', 'C'], ['L', 'S', 'T', 'R', 'D'],
                  ['S', 'T', 'R', 'D'], ['T', 'R', 'D'], ['R', 'D'],
                  ['C'], ['D']]
direction['S'] = [ ['A', 'Q', 'S', 'E'], ['B', 'Q', 'S', 'F'],
                  ['K', 'S', 'F'], ['L', 'S', 'E'],
                  ['M', 'R', 'Q', 'S', 'E'], ['N', 'R', 'Q', 'S', 'F'],
                  ['R', 'Q', 'S', 'E'], ['Q', 'S', 'E'], ['S', 'E'],
                  ['E'], ['F']]

# COPY OF THE POSSIBLE DIRECTIONS (otherwise car/ambu share the same pointer :s)
direction2 = dict()
direction2['W'] = [ ['A', 'Q', 'I'], ['B','Q','J'],
                  ['M', 'R', 'Q', 'I'], ['N','R','Q','J'],
                  ['G', 'T', 'R', 'Q', 'J'], ['H', 'T', 'R', 'Q', 'I'],
                  ['I'], ['J'],
                  ['Q', 'I'], ['R', 'Q', 'I'], ['T', 'R', 'Q', 'I']]
direction2['E'] = [ ['B', 'Q', 'S', 'T', 'O'], ['A', 'Q', 'S', 'T', 'P'],
                  ['K', 'S', 'T', 'O'], ['L', 'S', 'T', 'P'],
                  ['G', 'T', 'O'], ['H', 'T', 'P'],
                  ['T', 'P'], ['S', 'T', 'P'], ['Q', 'S', 'T', 'P'],
                  ['E'], ['F']]
direction2['N'] = [ ['G', 'T','R', 'C'], ['H', 'T', 'R', 'D'],
                  ['M', 'R', 'D'], ['N', 'R', 'C'],
                  ['K', 'S', 'T', 'R', 'C'], ['L', 'S', 'T', 'R', 'D'],
                  ['S', 'T', 'R', 'D'], ['T', 'R', 'D'], ['R', 'D'],
                  ['C'], ['D']]
direction2['S'] = [ ['A', 'Q', 'S', 'E'], ['B', 'Q', 'S', 'F'],
                  ['K', 'S', 'F'], ['L', 'S', 'E'],
                  ['M', 'R', 'Q', 'S', 'E'], ['N', 'R', 'Q', 'S', 'F'],
                  ['R', 'Q', 'S', 'E'], ['Q', 'S', 'E'], ['S', 'E'],
                  ['E'], ['F']]

```



```

# Calculate the path to destination for the ambulance
ambuPathSeg = None
painter.setPen(Qt.white)
for path in direction2[ambuDest]:
    if path[0]==ambuSegment:
        ambuPathSeg = path
if ambuPathSeg is None:
    QMessageBox.critical(self, 'Error', 'The ambu cannot go there!')
    return
else:
    #print ambuPathSeg
    if len(ambuPathSeg)>1:
        del ambuPathSeg[0]
    ambuPath = []
    previous_point = QPoint(ambuX, ambuY)
    for seg in ambuPathSeg:
        new_point = QPoint(self.xSeg[seg],self.ySeg[seg])
        ambuPath.append([previous_point, new_point])
        previous_point = new_point
    #print ambuPath
    for line in ambuPath:
        #print line
        painter.drawLine(*line)

# Calculate the path to destination for the car
carPathSeg = None
painter.setPen(dashPenRed)
for path in direction[carDest]:
    if path[0]==carSegment:
        carPathSeg = path
if carPathSeg is None:
    QMessageBox.critical(self, 'Error', 'The car cannot go there!')
    return
else:
    #print carPathSeg
    if len(carPathSeg)>1:
        del carPathSeg[0]
    carPath = []
    previous_point = QPoint(carX, carY)
    for seg in carPathSeg:
        new_point = QPoint(self.xSeg[seg],self.ySeg[seg])
        carPath.append([previous_point, new_point])
        previous_point = new_point

```

```

    #print carPath
    for line in carPath:
        #print line
        painter.drawLine(*line)

# =====
# = PERFORM THE ALGORITHM =
# =====

# distance of the first line of the car and the ambu
carFirstLineDist = dist(*carPath[0])
ambuFirstLineDist = dist(*ambuPath[0])

# If the car has already passed the intersection
# or is at the intersection
if len(carPathSeg)==1 or atIntersection(carSegment):
    # and the ambulance also
    if sameWay(carSegment, ambuSegment):
        # and even at the same segment
        if carSegment==ambuSegment:
            # if the car is ahead
            if carFirstLineDist < ambuFirstLineDist:
                # and can change lane, we ask to change lane
                if multipleLanes:
                    naviMessage = 'Please change lane, the ambulance is behind you'
                # else if we have the same destination, change the car destination
                elif carDest==ambuDest:
                    naviMessage = 'Please change your way to %s' %\
                        redirect(carSegment,carDest)
                else: # else there is nothing we can do except continuing
                    naviMessage = 'Ambulance is behind you. Stick to your plan.'
            # else, we are in the same way, but not the same segment
            elif carFirstLineDist < ambuFirstLineDist:
                # if we are ahead, give priority to the ambulance
                naviMessage = 'Stay in this lane and \
                    slow down to let the ambulance go first.'
            # else, we should see it, no need to worry the driver more
        # else, we are not in the same way
    elif ambuDest==carDest and len(carPathSeg) < len(ambuPathSeg):
        # if the car is coming this way, let the driver know it
        naviMessage = 'Ambulance will come this way.'
    # else, there is nothing special to do
    # (we are at the end segment and the ambu is not coming)

```

```

# Else, if we are in the same way (but not the last one)
elif sameWay(carSegment, ambuSegment):
    # and the same segment
    if ambuSegment==carSegment:
        # if the car is ahead
        if carFirstLineDist < ambuFirstLineDist:
            # and can change lane, advise the driver to do so
            if multipleLanes:
                naviMessage = 'Please change lane, slow down, \
                    and let the ambulance go first.'
                if carDest==ambuDest:
                    naviMessage +='\nIf not possible, please change your way to %s'\
                        % redirect(carSegment,carDest)
            # else, if we have the same destination,
            # change our way to the destination
            elif carDest==ambuDest:
                naviMessage = 'Please change your way to %s'\
                    % redirect(carSegment,carDest)
            else: # else there is nothing we can do except continuing
                naviMessage = 'Ambulance is behind you. Stick to your plan.'
        # else, we are in the same way but not the same segment
    elif carFirstLineDist < ambuFirstLineDist:
        #if we are ahead, let the driver know to stay in this lane but to slow down
        naviMessage='Stay in this lane and slow down to let the ambulance go first.'
        # else there is nothing special to do
        # (same way, different segment, behind the ambu)

# Else, if the ambu is at its last segment
# or at the intersection before its last segment
# there is nothing special
elif len(ambuPathSeg)==1 or atIntersection(ambuSegment):
    naviMessage = 'DISPLAY INFORMATION ONLY (2)'

# else, we are in different ways and both going to cross the intersection
# check if our routes intersect
elif routesIntersect(ambuPathSeg, carPathSeg):
    naviMessage = 'STOP at the next intersection and let the ambulance go first.'
# else, we are in different ways and are not going to cross our routes,
# just display info
else:
    naviMessage = 'DISPLAY INFORMATION ONLY (3)'

self.message.emit(naviMessage)

```

```

painter.fillRect(QRectF(carX, carY, 10,10), Qt.red)
painter.fillRect(QRectF(ambuX, ambuY, 10,10), Qt.white)

```

```

class AlgorithmValidationWindow(QMainWindow):
    def __init__(self):
        QMainWindow.__init__(self)
        mainWidget = QWidget()
        self.setCentralWidget(mainWidget)
        centralLayout = QHBoxLayout()
        #centralLayout.setSpacing(0)
        centralLayout.setAlignment(Qt.AlignTop)
        self.resize(500,400)
        mainWidget.setLayout(centralLayout)

        # left part
        leftWidget = QGroupBox(mainWidget, "Visualisation")
        leftWidget.setLayout(QVBoxLayout())
        self.image = AlgorithmValidationWidget()
        self.image.setMinimumHeight(300)
        self.image.setMinimumWidth(300)
        self.image.message.connect(self.updateNavi)
        leftWidget.layout().addWidget(self.image)
        self.carNaviLabel = QLabel()
        self.carNaviLabel.setAlignment(Qt.AlignCenter)
        self.carNaviLabel.setFont(QFont("Century Gothic", 10))
        self.carNaviLabel.setText("CAR NAVIGATION SYSTEM")
        leftWidget.layout().addWidget(self.carNaviLabel)
        centralLayout.addWidget(leftWidget)

        # right part
        rightWidget = QWidget(mainWidget)
        rightWidget.setLayout(QVBoxLayout())
        # CAR
        carBox = QGroupBox("Car settings")
        carBox.setLayout(QVBoxLayout())
        self.carSegmentOption = SimpleComboboxOption('carseg', 'Segment', 0, False,
                                                    'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',
                                                    'I', 'J', 'K', 'L', 'M', 'N', 'O',
                                                    'P', 'Q', 'R', 'S', 'T')
        self.carDiffOption = SimpleComboboxOption('cardif', 'Difference', 1, False,
                                                  '+1', '0', '-1')
        self.carDestOption = SimpleComboboxOption('cardest', 'Destination', 2, False,
                                                  'N', 'S', 'W', 'E')
        carBox.layout().addWidget(self.carSegmentOption)

```

```

carBox.layout().addWidget(self.carDiffOption)
carBox.layout().addWidget(self.carDestOption)
rightWidget.layout().addWidget(carBox)
# AMBU
ambuBox = QGroupBox("Ambu settings")
ambuBox.setLayout(QVBoxLayout())
self.ambuSegmentOption = SimpleComboboxOption('ambuseg','Segment',6, False,
                                                'A','B','C','D', 'E', 'F', 'G', 'H',
                                                'I', 'J', 'K', 'L', 'M', 'N', 'O',
                                                'P', 'Q', 'R', 'S', 'T')
self.ambuDiffOption = SimpleComboboxOption('ambudif','Difference',1, False,
                                             '+1','0','-1')
self.ambuDestOption = SimpleComboboxOption('ambudest','Destination',2, False,
                                             'N', 'S', 'W', 'E')

ambuBox.layout().addWidget(self.ambuSegmentOption)
ambuBox.layout().addWidget(self.ambuDiffOption)
ambuBox.layout().addWidget(self.ambuDestOption)
rightWidget.layout().addWidget(ambuBox)
# Button
self.applyButton = QPushButton('APPLY')
self.applyButton.clicked.connect(self.applySettings)
rightWidget.layout().addWidget(self.applyButton)
centralLayout.addWidget(rightWidget)
def updateNavi(self, msg):
    #self.carNaviLabel.setText('<div style="background-color:black">
    # <font color="red">%s</font></div>%msg.replace('\n','<br />'))
    self.carNaviLabel.setText('<font color="red">%s</font>%msg.replace('\n','<br />'))
def applySettings(self):
    self.image.placeCars(self.carSegmentOption.getName(), self.carDiffOption.getName(),
                        self.carDestOption.getName(),self.ambuSegmentOption.getName(),
                        self.ambuDiffOption.getName(),self.ambuDestOption.getName())

mw = AlgorithmValidationWindow()
mw.setWindowTitle('Nishimura Lab | TC4EV Algo Validation')
mw.show()

sys.exit(app.exec_())

```

APPENDIX C

HIGHWAY SIMULATIONS SOURCE CODE

```
__author__ = "Thomas Dall'Agnese"

from PySide.QtGui import QMainWindow, QTabWidget, QVBoxLayout, QGroupBox, QPushButton, \
    QProgressBar, QMessageBox, QGridLayout, QTextBrowser, QFont, QPixmap, QImage
from PySide.QtCore import Slot, QThreadPool, QThread, QSize
from helpers import *
from wafThread import *
from datetime import datetime
import json

class HighwaySimulatorGui(QMainWindow):
    def __init__(self):
        super(HighwaySimulatorGui, self).__init__()
        centralTab = QTabWidget()
        mainWidget = QWidget()
        self.resultsWidget = HighwayAnalyzeWidget()
        centralTab.addTab(mainWidget, 'Run')
        centralTab.addTab(self.resultsWidget, 'Analyze')
        self.setCentralWidget(centralTab)
        centralLayout = QVBoxLayout()
        #centralLayout.setSpacing(0)
        centralLayout.setAlignment(Qt.AlignTop)
        gridWidget = QWidget()
        gridLayout = QGridLayout()
        gridLayout.setSpacing(0)
        gridWidget.setLayout(gridLayout)
        mainWidget.setLayout(centralLayout)
        self.options = dict()
        # GENERAL
        generalGroup = QGroupBox('General Settings')
        generalGroup.setLayout(QVBoxLayout())
        generalGroup.layout().setSpacing(0)
        self.pathOption = SimpleOption('path', 'Output Path',
                                      '/home/thomas/Dropbox/Keio/research/results/')
        generalGroup.layout().addWidget(self.pathOption)
        self.scenarioOption = SimpleComboboxOption('scenario', 'Scenario', 1, False,
                                                  'vanet-highway-test-thomas', 'vanet-highway-scenario2')
        self.scenarioOption.combo.currentIndexChanged[int].connect(self.scenarioChanged)
        generalGroup.layout().addWidget(self.scenarioOption)
        self.options['time'] = SimpleSpinOption('time', 'Simulation Time (sec.)', 1500, True)
```

```

self.options['time'].setRange(0,3000)
self.options['mix'] = SimpleSpinOption('mix',
                                     'Percentage of cars compare to trucks (%)',80,True)
self.options['mix'].setRange(0,100)
self.options['gap'] = SimpleSpinOption('gap','Average Gap (m.)',5)
self.options['gap'].setRange(1,2000)
self.options['lane'] = SimpleSpinOption('lane','Number of Lanes',2,True)
self.options['lane'].setRange(2,4)
self.options['spl'] = SimpleSpinOption('spl','Speed Limit (km/h)',130,True)
self.options['spl'].setRange(1,200)
for widget in ('time','mix','gap','lane','spl'):
    generalGroup.layout().addWidget(self.options[widget])
gridLayout.addWidget(generalGroup,0,0)
# TRAFFIC
trafficGroup = QGroupBox('Traffic Settings')
trafficGroup.setLayout(QVBoxLayout())
trafficGroup.layout().setSpacing(0)
# m/s = (km/h)*1000/3600
self.options['vell'] = SimpleSpinOption('vell','Average Speed (km/h)',105,True)
self.options['vell'].setRange(5,150)
self.options['dis'] = SimpleComboboxOption('dis','Speed Distribution Model',3,
                                           False, 'Uniform','Exponential','Normal','Log Normal')
self.options['spstd'] = SimpleSpinOption('spstd','Speed Distribution Variance',1.0)
self.options['spstd'].setRange(0,50)
self.options['flow1'] = SimpleSpinOption('flow1','Traffic Flow Mean (veh/s)',1.0)
self.options['flow1'].setRange(0.1,50.0)
self.options['std1'] = SimpleSpinOption('std1','Traffic Flow Variance',0.8)
self.options['std1'].setRange(0.1,50.0)
self.options['maxflow'] = SimpleSpinOption('maxflow','Traffic Maximum Flow (veh/s)',
                                           ,5)
self.options['maxflow'].setRange(0.1,50.0)
# Scenar 2
self.options['avgdist'] = SimpleSpinOption('avgdist','Average Distance (m)',100)
self.options['avgdist'].setRange(1,10000)
self.options['avgspeed'] = SimpleSpinOption('avgspeed','Average Speed (km/h)',105)
self.options['avgspeed'].setRange(1,10000)
self.options['despeed'] = SimpleSpinOption('despeed','Desired Speed (km/h)',130)
self.options['despeed'].setRange(1,10000)
self.options['ambumaxspeed'] = SimpleSpinOption('ambumaxspeed',
                                                'Ambu Max Speed (km/h)',165)
self.options['ambumaxspeed'].setRange(1,10000)
self.options['ambuunitspeed'] = SimpleSpinOption('ambuunitspeed',
                                                'Ambu Initial Speed (km/h)',130)
self.options['ambuunitspeed'].setRange(1,10000)

```

```

for widget in ('vell','dis','spstd','flow1','std1','maxflow',
              'avgdist', 'avgspeed', 'despeed', 'ambumaxspeed', 'ambuinitsspeed'):
    trafficGroup.layout().addWidget(self.options[widget])
self.scenarioChanged(self.scenarioOption.combo.currentIndex())
gridLayout.addWidget(trafficGroup,0,1)
# VANET
vanetGroup = QGroupBox('VANET Settings')
vanetGroup.setLayout(QVBoxLayout())
vanetGroup.layout().setSpacing(0)
#
    self.options['prate'].setRange(0,100)
self.options['prate'] = 0 # start with 0
self.options['pw'] = SimpleSpinOption('pw','Transmission Power (dBm)',21.5)
self.options['pw'].setRange(10,50)
#for widget in ('prate','pw'):
for widget in ('pw',):
    vanetGroup.layout().addWidget(self.options[widget])
gridLayout.addWidget(vanetGroup,1,0)
# BATCH SETTINGS
batchGroup = QGroupBox("Batch Settings")
batchGroup.setLayout(QVBoxLayout())
self.gapPrateOption = SimpleSpinOption('gapPrate',
                                       'VANET percentage rate gap', 10, integer=True)
self.sameSimuTimesOption = SimpleSpinOption('sameSimuTimes',
                                             'How many times the same simulation', 100, integer=True)
batchGroup.layout().setSpacing(0)
batchGroup.layout().addWidget(self.gapPrateOption)
batchGroup.layout().addWidget(self.sameSimuTimesOption)
gridLayout.addWidget(batchGroup,1,1)
# START SIMU
centralLayout.addWidget(gridWidget)
self.startButton = QPushButton('START')
self.startButton.clicked.connect(self.startSimu)
centralLayout.addWidget(self.startButton)
self.progressBar = QProgressBar()
centralLayout.addWidget(self.progressBar)
self.shutdownWhenDone = QCheckBox('Shutdown when done')
self.actionWhenDone = QComboBox()
self.actionWhenDone.addItem(['When finished... do nothing',
                             'When finished... shutdown the computer',
                             'When finished... Re-run the simulations!'])
self.actionWhenDone.setCurrentIndex(int(QSettings().value('actionWhenDone', 0)))
centralLayout.addWidget(self.actionWhenDone)
self.infoLabel = QLabel()
centralLayout.addWidget(self.infoLabel)

```



```

# LOG
self.logText = QTextBrowser()
self.logText.setFont(QFont('Century Gothic', 7))
centralLayout.addWidget(self.logText)
self.setWindowTitle('Nishimura Lab | Highway Simulation')
#self.resize(520, len(self.options)*60+100)
#self.resultFile=open('/home/thomas/Dropbox/Keio/research/results/summary.txt','a')
#self.resultFile=os.path.join(self.pathOption.getValue(), 'summary.txt')
self.logFile = os.path.join(self.pathOption.getValue(),
                             'results_'+os.uname()[1]+'.log')

@Slot(int)
def scenarioChanged(self, index):
    #print index
    scenar1options = ['gap', 'vell', 'dis', 'flow1', 'std1', 'spstd', 'maxflow']
    scenar2options = ['avgdist', 'avgspeed', 'despeed', 'ambumaxspeed', 'ambuunitspeed']
    if index==0: # first scenario
        scenar1 = True
        scenar2 = False
    else:
        scenar1 = False
        scenar2 = True
    for option in scenar1options:
        self.options[option].setVisible(scenar1)
    for option in scenar2options:
        self.options[option].setVisible(scenar2)
def log(self, txt):
    toLog = '%s | %s' % (datetime.now(), txt)
    with open(self.logFile, 'a') as logFile:
        logFile.write(toLog+'\n')
    self.logText.append(toLog)
def blockUi(self):
    self.startButton.setEnabled(False)
    self.scenarioOption.setEnabled(False)
    for option in self.options:
        if option!='prate':
            self.options[option].setEnabled(False)
def releaseUi(self):
    self.startButton.setEnabled(True)
    self.scenarioOption.setEnabled(True)
    self.startButton.setText('START')
    for option in self.options:
        if option!='prate':
            self.options[option].setEnabled(True)
def startSimu(self):

```

```

self.log("=== SIMULATIONS START ===")
self.logText.clear()
self.startTime = datetime.now()
self.simulations = []
#self.nextPrate = 0
self.gapPrate = self.gapPrateOption.getValue()
self.sameSimuTimes = self.sameSimuTimesOption.getValue()
#self.nextSimu = 0
self.blockUi()
self.simulationsDone = 0
#output = self.pathOption.getValue() + dateToFilename(d) + '_results.txt'
pRate = 0
self.simulationsTotal = 0
# print 'sameSimuTimes: %d'%self.sameSimuTimes
# print 'gapPrate: %d'%self.gapPrate
while pRate <= 100:
    simu = 0
    self.options['prate'] = pRate
    while simu<self.sameSimuTimes:
        waf = WafThread(self.options, self.pathOption.getValue(),
                        self.scenarioOption.getName())
        waf.setAutoDelete(True)
        waf.simuDone.connect(self.wafDone)
        self.simulations.append(waf)
        QThreadPool.globalInstance().start(waf)
        self.simulationsTotal += 1
        simu += 1
    pRate += self.gapPrate
runningSimulations = 'Running %d Simulations...' % self.simulationsTotal
self.startButton.setText(runningSimulations)
self.log(runningSimulations)
self.progressBar.setRange(0,self.simulationsTotal)
self.progressBar.setValue(0)
# 300 seconds per task average
roughTime = self.simulationsTotal*self.resultsWidget.averageSimulationTime\
            /QThreadPool.globalInstance().maxThreadCount()
self.infoLabel.setText('Rough time estimation: %s' % formatTimeLeft(roughTime))
@Slot(str)
def wafDone(self, outputPath):
    #print 'thread done!\nReceived:'
    self.simulationsDone += 1
    simulationsLeft = self.simulationsTotal-self.simulationsDone
    with open(outputPath,'r') as out:
        r = json.loads(out.read())

```

```

out.close()
if 'timeToReachDest' in r['results']:
    try:
        result = "%3.3d\t%f" % (int(r['settings']['prate']),
                                float(r['results']['timeToReachDest']))
    except TypeError, e:
        result = 'PYTHON ERROR: %s | File: %s' % (e, outputPath)
        print result
        print r
    else:
        result = 'ERROR: No timeToReachDest (%s)' % outputPath
self.startButton.setText('%d simulations left...' % simulationsLeft)
self.log(result)
# with open(self.resultFile, 'a') as summary:
#     summary.write('%s | %s\n' % (datetime.now(), result))
#     summary.close()
self.progressBar.setValue(self.simulationsDone)
if self.simulationsDone==self.simulationsTotal:
    QThreadPool.globalInstance().waitForDone()
    del self.simulations[:]
    self.releaseUi()
    if self.actionWhenDone.currentIndex()==1:
        self.saveSettings()
        from time import sleep
        print 'shutdown in 20 seconds...'
        sleep(20)
        os.system('sudo halt')
    elif self.actionWhenDone.currentIndex()==2:
        self.startSimu()
else:
    # calculate estimated time left
#     percentage_done = 1.0*self.simulationsDone/self.simulationsTotal
#     done_in = (datetime.now()-self.startTime).seconds
#     timeLeft = done_in*((1.0/percentage_done)-1.0)
# v2: average time per simulation * nb of simulations left
averageTimePerSimulation = (datetime.now()-self.startTime).seconds\
                                /self.simulationsDone
timeLeft = averageTimePerSimulation * simulationsLeft
#print "%f perc done in %d seconds => %d seconds left"\
# % (percentage_done, done_in, timeLeft)
formattedTimeLeft = formatTimeLeft(timeLeft)
self.infoLabel.setText("Estimated time left: %s" % formattedTimeLeft)
def saveSettings(self):
    for setting in self.options:

```

```

        if setting != 'prate':
            self.options[setting].save()
self.pathOption.save()
self.scenarioOption.save()
QSettings().setValue('actionWhenDone',self.actionWhenDone.currentIndex())
QSettings().setValue('sameSimuTimes',self.sameSimuTimesOption.getValue())
QSettings().setValue('gapPrate',self.gapPrateOption.getValue())
self.resultsWidget.saveSettings()
def closeEvent(self, *args, **kwargs):
    self.saveSettings()
    try:
        for waf in self.simulations:
            del waf
        del self.simulations[:]
    except:
        pass
    super(HighwaySimulatorGui,self).closeEvent(*args, **kwargs)

```

```

class HighwayAnalyzeWidget(QWidget):
    def __init__(self):
        super(HighwayAnalyzeWidget,self).__init__()
        self.averageSimulationTime = 325.0
        mainLayout = QVBoxLayout()
        mainLayout.setAlignment(Qt.AlignTop|Qt.AlignHCenter)
        mainLayout.setSpacing(0)
        self.setLayout(mainLayout)
        topWidget = QWidget()
        self.topLayout = QHBoxLayout()
        self.topLayout.setSpacing(0)
        topWidget.setLayout(self.topLayout)
        topLeftWidget = QWidget()
        self.topLeftLayout = QVBoxLayout()
        topLeftWidget.setLayout(self.topLeftLayout)
        self.setupTopGroup()
        self.setupFilterGroup()
        self.topLayout.addWidget(topLeftWidget)
        self.layout().addWidget(topWidget)
        # Button and log
        buttonAndLogWidget = QWidget()
        buttonAndLogWidget.setLayout(QVBoxLayout())
        # Analyze the results BUTTON
        self.analyzeResultsButton = QPushButton('Analyze the results')
        self.analyzeResultsButton.clicked.connect(self.analyzeResults)
        self.analyzeResultsButton.setEnabled(False)

```

```

buttonAndLogWidget.layout().addWidget(self.analyzeResultsButton)
# LOG
self.logText = QTextBrowser()
self.logText.setFont(QFont('Century Gothic', 7))
self.logText.setWordWrapMode(QTextOption.NoWrap)
buttonAndLogWidget.layout().addWidget(self.logText)
self.topLayout.addWidget(buttonAndLogWidget)
self.results = []
self.logFile = os.path.join(self.resultsPath(), 'analyze_'+os.uname()[1]+'.log')
# Image
self.picLabel = QLabel()
self.picLabel.setAlignment(Qt.AlignHCenter)
#self.picLabel.resize(800,600)
self.picLabel.setMinimumSize(800,600)
self.layout().addWidget(self.picLabel)
def setupTopGroup(self):
    topGroup = QGroupBox("Simulation Results")
    self.resultsPathLineEdit = SimpleOption('resultsPath','Results Path',
                                           '/home/thomas/Dropbox/Keio/research/results/')
    topGroupLayout = QVBoxLayout()
    topGroupLayout.setSpacing(0)
    topGroupLayout.setAlignment(Qt.AlignTop)
    topGroupLayout.addWidget(self.resultsPathLineEdit)
    self.loadResultsLabel = QLabel()
    self.loadResultsLabel.setAlignment(Qt.AlignHCenter)
    topGroupLayout.addWidget(self.loadResultsLabel)
    self.loadResultsButton = QPushButton("Load the results")
    self.loadResultsButton.clicked.connect(self.loadResults)
    topGroupLayout.addWidget(self.loadResultsButton)
    topGroup.setLayout(topGroupLayout)
    self.topLeftLayout.addWidget(topGroup)
def resultsPath(self):
    return self.resultsPathLineEdit.getValue()
def setupFilterGroup(self):
    filterGroup = QGroupBox('Filter the results')
    filterGroupLayout = QVBoxLayout()
    filterGroupLayout.setSpacing(0)
    filterGroupLayout.setAlignment(Qt.AlignTop)
    # Distribution Model
    self.filterDistribution = SimpleComboboxOption('dis','Speed Distribution Model',3,
                                                  True, 'Uniform','Exponential','Normal','Log Normal')
    filterGroupLayout.addWidget(self.filterDistribution)
    self.filterDistribution.setVisible(False)
    # Number of results per point

```

```

self.filterNb = SimpleSpinOption('simuNbMin', 'Minimum results for a given setting',
                                10, integer=True, checkable=True)

self.filterNb.checkBox.setChecked(True)
filterGroupLayout.addWidget(self.filterNb)
# Filter the date
dateWidget = QWidget()
dateWidget.setLayout(QHBoxLayout())
self.filterDate = QCheckBox('After')
self.filterDate.setChecked(QSettings().value('filterDate', '0')== '1')
dateWidget.layout().addWidget(self.filterDate)
self.filterDateYear = QComboBox()
for m in xrange(2010, 2012):
    self.filterDateYear.addItem(str(m))
self.filterDateYear.setCurrentIndex(int(QSettings().value('dateYear', 1)))
dateWidget.layout().addWidget(self.filterDateYear)
dateWidget.layout().addWidget(QLabel('Year'))
self.filterDateMonth = QComboBox()
for m in xrange(1, 13):
    self.filterDateMonth.addItem(str(m))
self.filterDateMonth.setCurrentIndex(int(QSettings().value('dateMonth', 4)))
dateWidget.layout().addWidget(self.filterDateMonth)
dateWidget.layout().addWidget(QLabel('Month'))
self.filterDateDay = QComboBox()
for d in xrange(1, 32):
    self.filterDateDay.addItem(str(d))
self.filterDateDay.setCurrentIndex(int(QSettings().value('dateDay', 0)))
dateWidget.layout().addWidget(self.filterDateDay)
dateWidget.layout().addWidget(QLabel('Day'))
filterGroupLayout.addWidget(dateWidget)
filterGroup.setLayout(filterGroupLayout)
self.topLeftLayout.addWidget(filterGroup)
# Filter the scenario
self.filterScenar = SimpleComboboxOption('scenar', 'Scenario', 1, True,
                                          'vanet-highway-test-thomas', 'vanet-highway-scenario2')
filterGroupLayout.addWidget(self.filterScenar)
# Filter gap
self.filterGap = SimpleSpinOption('avgdistanalyze', 'Average Distance (m)', 100,
                                  checkable=True)
filterGroupLayout.addWidget(self.filterGap)
def loadResults(self):
    self.loadResultsButton.setText('Loading the results...')
    self.loadResultsButton.setEnabled(False)
    self.results = []
    self.resultsThread = LoadResults(resultsPath=self.resultsPath())

```

```

        self.resultsThread.done.connect(self.loadResultsDone)
        self.resultsThread.error.connect(self.loadResultsError)
        self.resultsThread.start()
@Slot(str)
def loadResultsError(self, errorMsg):
    QMessageBox.critical(self, 'Error!', errorMsg)
    self.analyzeResultsButton.setEnabled(False)
@Slot(dict)
def loadResultsDone(self, results):
    self.results = results
    resultsNb = len(self.results)
    msg= 'Found %d Simulation Results\n(%d invalid files, %d simulations failed)' %\
(resultsNb, self.resultsThread.fileUnloadable, self.resultsThread.simulationFailed)
    self.loadResultsLabel.setText(msg)
    #self.calculateAverageSimulationTime()
    self.loadResultsButton.setText('Reload the results')
    self.loadResultsButton.setEnabled(True)
    self.analyzeResultsButton.setEnabled(True)
    if resultsNb>0:
        self.analyzeResults()
def calculateAverageSimulationTime(self):
    totalTime = 0
    nb = 0
    for r in self.results:
        if 'simulationTime' in r:
            #print 'simu time: %s' % r['simulationTime']
            totalTime += int(r['simulationTime'])
            nb += 1
            #print 'from %s: [%3.3f,%3.3f]' % (r['filename'], r['settings']['prate'],
            # r['results']['timeToReachDest'])
    if nb<=0:
        errorMsg= 'No simulation found with simulationTime'
        QMessageBox.critical(self, 'Error!', errorMsg)
        self.averageSimulationTime = 290.0
    else:
        self.averageSimulationTime= totalTime/nb
        self.loadResultsLabel.setText(self.loadResultsLabel.text()+
            '\nAverage simulation time: %3.0f'%self.averageSimulationTime)
def fil(self, r):
    return self.checkDis(r) and self.checkDate(r) and self.checkGap(r)\
        and self.checkScenario(r)
def checkDis(self,r):
    return (not self.filterDistribution.checkBox.isChecked())\
        or r['settings']['dis']==self.filterDistribution.getValue()

```

```

def checkDate(self, r):
    return (not self.filterDate.isChecked())\
           or (r['date'] >= datetime(int(self.filterDateYear.currentText()),
                                     int(self.filterDateMonth.currentText()),
                                     int(self.filterDateDay.currentText())))

def checkScenario(self, r):
    return (not self.filterScenar.checkBox.isChecked())\
           or r['scenario']==self.filterScenar.getName()

def checkGap(self, r):
    return (not self.filterGap.checkBox.isChecked())\
           or float(r['settings']['avgdist'])==float(self.filterGap.getValue())

def analyzeResults(self):
    self.saveSettings()
    if len(self.results)<=0:
        QMessageBox.critical(self, 'Error!', 'No results loaded :s')
        return
    self.log("=== ANALYZING RESULTS ===")
    self.logText.clear()
    p=0
    meanValues = {}
    uniqueName = dateToFilename()
    gnuPlotDataFile =os.path.join(self.resultsPath(), 'graphs/' + uniqueName+'.dat')
    with open(gnuPlotDataFile, 'w') as data:
        while p<=100:
            nb = 0
            totalTimeToReachDest = 0
            totalSimulationTime = 0
            for result in filter(lambda x: x['settings']['prate']==p
                                and self.fil(x), self.results):
                totalSimulationTime += float(result['simulationTime'])
                totalTimeToReachDest += float(result['results']['timeToReachDest'])
                nb += 1
            if self.filterNb.checkBox.isChecked():
                minNb = self.filterNb.getValue()
            else:
                minNb = 0
            if nb>minNb:
                meanSimulationTime = float(1.0*totalSimulationTime/nb)
                meanTimeToReachDest = float(1.0*totalTimeToReachDest/nb)
                meanValues[p] = {'prate':p,
                                'simuLationTime':meanSimulationTime,
                                'simulationsNb':nb,
                                'timeToReachDest':meanTimeToReachDest}
                self.log(meanValues[p])

```



```

        toPlot = '%s %s' % (p, meanValues[p]['timeToReachDest'])
        #print toPlot
        data.write(toPlot+'\n')
    p += 1
    data.close()
if len(meanValues)>0:
    outputPic = 'graphs/' + uniqueName
    s = subprocess.Popen(['./toPlot.sh', outputPic, gnuPlotDataFile],
                        cwd=self.resultsPath())

    s.wait()
    outputPicPath = os.path.join(self.resultsPath(),outputPic+'.svg')
    pic = QImage(outputPicPath)
    #pic = pic.scaled(QSize(640,480))
    self.picLabel.setPixmap(QPixmap(pic))
    #os.system(os.path.join(self.resultsPath(),'toPlot.sh'))
else:
    QMessageBox.critical(self, 'Error!', 'No simulation satisfies the criteria...')
def log(self, txt):
    toLog = str(txt)
    toLogFile = '%s | %s' % (datetime.now(), txt)
    with open(self.logFile, 'a') as logFile:
        logFile.write(toLogFile+'\n')
    self.logText.append(toLog)
def saveSettings(self):
    QSettings().setValue('simuNbMin', self.filterNb.getValue())
    QSettings().setValue('avgdistanalyze', self.filterGap.getValue())
    QSettings().setValue('scenar', self.filterScenar.getValue())
    QSettings().setValue('dis', self.filterDistribution.getValue())
    QSettings().setValue('dateDay', self.filterDateDay.currentIndex())
    QSettings().setValue('dateMonth', self.filterDateMonth.currentIndex())
    QSettings().setValue('dateYear', self.filterDateYear.currentIndex())
    if self.filterDate.isChecked():
        QSettings().setValue('filterDate', '1')
    else:
        QSettings().setValue('filterDate', '0')

class LoadResults(QThread):
    done = Signal(list)
    error = Signal(str)
    def __init__(self, resultsPath):
        super(LoadResults,self).__init__()
        self.resultsPath = resultsPath
        self.fileUnloadable = 0
        self.simulationFailed = 0

```

```

self.results = []
def run(self):
    results = []
    self.fileUnloadable = 0
    self.simulationFailed = 0
    if os.path.exists(self.resultsPath):
        tmpdir = tempfile.mkdtemp(prefix='results_tmpdir',dir=self.resultsPath)
        # temporary extract the tar.gz files in this folder
        for filename in filter(lambda x: x.endswith('.tar.gz')
                               and x.startswith('results'), os.listdir(self.resultsPath)):
            tar = tarfile.open(os.path.join(self.resultsPath,filename), "r:gz")
            tar.extractall(tmpdir)
            tar.close()
    for path, subdirs, files in os.walk(self.resultsPath):
        for name in filter(lambda x: x.endswith('.txt')
                           and not x.endswith('_ambu.txt'), files):
            #for name in files:
                filePath = os.path.join(path,name)
                with open(filePath) as result:
                    try:
                        r = json.loads(result.read())
                    except:
                        #print 'errra with file %s' % name
                        self.fileUnloadable += 1
                    else:
                        #print r['date']
                        if 'results' in r and 'timeToReachDest' in r['results']\
                            and 'succeed' in r and r['succeed'] == 1:
                            r['filename'] = name
                            fileDate=datetime.fromtimestamp(os.path.getmtime(filePath))
                            r['date'] = fileDate
                            if 'avgdist' not in r['settings']:
                                r['settings']['avgdist'] = 100
                            if 'scenario' not in r:
                                if fileDate >= datetime(2011, 6, 17):
                                    r['scenario'] = 'vanet-highway-scenario2'
                                else:
                                    r['scenario'] = 'vanet-highway-test-thomas'
                            print fileDate
                            print '--> %s' % r['scenario']
                            results.append(r)
                    else:
                        self.simulationFailed += 1
                result.close()

```

```
# remove the temporary folder
for filename in os.listdir(tempdir):
    os.remove(os.path.join(tempdir, filename))
os.rmdir(tempdir)
self.results = results
self.done.emit(results)
else:
    errorMsg = 'The directory containing the results (%s) does not exist' % \
        self.resultsPath
    self.error.emit(errorMsg)
```