Departamento de Informática de Sistemas y Computadores

Efficient Traffic Management in Urban Environments

Thesis submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy in Computer Science

By

Jorge Luis Zambrano Martínez

PhD Advisors:

Dr. Carlos Tavares Calafate
Dr. David Soler Fernández

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Valencia, Spain

September, 2019

Any intelligent fool can make things bigger and more complex... It takes a touch of genius – and a lot of courage to move in the opposite direction.

Prof. Albert Einstein

Acknowledgements

At the end of an arduous work as the development of a doctoral thesis it is inevitable to be assaulted by a very human egocentricity that leads you to concentrate most of the merit in the contribution in the scientific world. However, an objective analysis shows you immediately that the magnitude of this contribution would have been impossible without the participation of people and institutions that have made it easier for this work to reach a successful conclusion. Therefore, it is a real pleasure for me to use this space to be fair and consistent with them, expressing my thanks.

I want to express my sincere thanks to Dr. Carlos Tavares Calafate for his friendship as well as his meticulous support, guidance and active participation in the development of this thesis. I must emphasize, above all, his availability and patience that made our ever heated discussions beneficially rewarding both scientifically and personally.

Special thanks to Dr. David Soler for allowing this doctoral thesis to be developed within the mathematical framework of the different theories proposed.

Besides, I want to thank the Networking Research Group (GRC) professors, especially Dr. Pietro Manzoni, for opening me the laboratory doors, and allowing me to work in the group, and to Dr. Juan Carlos Cano for his support and corrections.

I also want to thank Dr. Thierry Gayraud from the research group Services and Architectures for Advanced Networks (SARA) at the Laboratory for Analysis and Architecture of Systems (LAAS) of the French National Centre for Scientific Research (CNRS), for receiving me during my internship.

A special thanks to GRC members, all of them, for their friendship, and for allowing me to share wonderful moments both inside and outside the laboratory, during the "Charlas On the Go" meetings, and in general on all GRC events.

To my whole family the deepest gratitude for encouraging me to face new challenges. Without their support, collaboration and inspiration, it would have

been impossible to undertake this pleasant experience in distant lands, and be an example always to follow. To my parents, Naisy Paulina and Jorge Eduardo, for their example of struggle, honesty, patience, courage and self-improvement, and to my little sister Naysi Cristina for her tenacity, self-improvement, intelligence and generosity.

Finally, I want to thank the Ecuatorian Republic through the "Secretaría de Educación Superior, Ciencia, Tecnología e Innovación" (SENESCYT), for granting me the scholarship to finance my studies.

Jorge Luis Zambrano Martínez
Valencia, Friday 13th September, 2019

Abstract

Currently, one of the main challenges that large metropolitan areas have to face is traffic congestion, which has become an important problem faced by city authorities. To address this problem, it becomes necessary to implement an efficient solution to control traffic that generates benefits for citizens, such as reducing vehicle journey times and, consequently, use of fuel, noise and environmental pollution. In fact, by properly analyzing traffic demand, it becomes possible to predict future traffic conditions, and to use that information for the optimization of the routes taken by vehicles. Such an approach becomes especially effective if applied in the context of autonomous vehicles, which have a more predictable behavior, thus enabling city management entities to mitigate the effects of traffic congestion and pollution by improving the traffic flow in a city in a fully centralized manner. Validating this approach typically requires the use of simulations, which should be as realistic as possible. However, achieving high degrees of realism can be complex when the actual traffic patterns, defined through an Origin/Destination (O-D) matrix for the vehicles in a city, are unknown, as occurs most of the times. Thus, the first contribution of this thesis is to develop an iterative heuristic for improving traffic congestion modeling; starting from real induction loop measurements made available by the City Hall of Valencia, Spain, we were able to generate an O-D matrix for traffic simulation that resembles the real traffic distribution. If it were possible to characterize the state of traffic by predicting future traffic conditions for optimizing the route of automated vehicles, and if these measures could be taken to preventively mitigate the effects of congestion with its related problems, the overall traffic flow could be improved. Thereby, the second contribution of this thesis was to develop a Traffic Prediction Equation to characterize the different streets of a city in terms of travel time with respect to the vehicle load, and applying logistic regression to those data to predict future traffic conditions. The third and last contribution of this thesis towards our envisioned traffic management paradigm was a route server capable of handling all the traffic in a city,

and balancing traffic flows by accounting for present and future traffic congestion conditions. Thus, we perform a simulation study using real data of traffic congestion in the city of Valencia, Spain, to demonstrate how the traffic flow in a typical day can be improved using our proposed solution. Experimental results show that our proposed solution, combined with frequent updating of traffic conditions on the route server, is able to achieve substantial improvements in terms of average travel speeds and travel times, both indicators of lower degrees of congestion and improved traffic fluidity.

Resumen

En la actualidad, uno de los principales desafíos a los que se enfrentan las grandes áreas metropolitanas es la congestión provocada por el tráfico, la cual se ha convertido en un problema importante al que se enfrentan las autoridades de cada ciudad. Para abordar este problema es necesario implementar una solución eficiente para controlar el tráfico que genere beneficios para los ciudadanos, como reducir los tiempos de viaje de los vehículos y, en consecuencia, el consumo de combustible, el ruido, y la contaminación ambiental. De hecho, al analizar adecuadamente la demanda de tráfico, es posible predecir las condiciones futuras del tráfico, y utilizar esa información para la optimización de las rutas tomadas por los vehículos. Este enfoque puede ser especialmente efectivo si se aplica en el contexto de los vehículos autónomos, que tienen un comportamiento más predecible, lo cual permite a los administradores de la ciudad mitigar los efectos de la congestión, como es la contaminación, al mejorar el flujo de tráfico de manera totalmente centralizada. La validación de este enfoque generalmente requiere el uso de simulaciones que deberían ser lo más realistas posible. Sin embargo, lograr altos grados de realismo puede ser complejo cuando los patrones de tráfico reales, definidos a través de una matriz de Origen/Destino (O-D) para los vehículos en una ciudad, son desconocidos, como ocurre la mayoría de las veces. Por lo tanto, la primera contribución de esta tesis es desarrollar una heurística iterativa para mejorar el modelado de la congestión de tráfico; a partir de las mediciones de bucle de inducción reales hechas por el Ayuntamiento de Valencia (España), pudimos generar una matriz O-D para la simulación de tráfico que se asemeja a la distribución de tráfico real. Si fuera posible caracterizar el estado del tráfico prediciendo las condiciones futuras del tráfico para optimizar la ruta de los vehículos automatizados, y si se pudieran tomar estas medidas para mitigar de manera preventiva los efectos de la congestión con sus problemas relacionados, se podría mejorar el flujo de tráfico en general. Por lo tanto, la segunda contribución de esta tesis es desarrollar una Ecuación de Predicción de Tráfico para caracterizar el comportamiento en las diferentes calles

de la ciudad en términos de tiempo de viaje con respecto al volumen de tráfico, y aplicar una regresión logística a esos datos para predecir las condiciones futuras del tráfico. La tercera y última contribución de esta tesis apunta directamente al nuevo paradigma de gestión de tráfico previsto, tratándose de un servidor de rutas capaz de manejar todo el tráfico en una ciudad, y equilibrar los flujos de tráfico teniendo en cuenta las condiciones de congestión del tráfico presentes y futuras. Por lo tanto, realizamos un estudio de simulación con datos reales de congestión de tráfico en la ciudad de Valencia (España), para demostrar cómo se puede mejorar el flujo de tráfico en un día típico mediante la solución propuesta. Los resultados experimentales muestran que nuestra solución, combinada con una actualización frecuente de las condiciones del tráfico en el servidor de rutas, es capaz de lograr mejoras sustanciales en términos de velocidad promedio y tiempo de trayecto, ambos indicadores de un menor grado de congestión y de una mejor fluidez del tráfico.

Resum

En l'actualitat, un dels principals desafiaments als quals s'enfronten les grans àrees metropolitanes és la congestió provocada pel trànsit, que s'ha convertit en un problema important al qual s'enfronten les autoritats de cada ciutat. Per a abordar aquest problema és necessari implementar una solució eficient per a controlar el trànsit que genere beneficis per als ciutadans, com reduir els temps de viatge dels vehicles i, en conseqüència, el consum de combustible, el soroll, i la contaminació ambiental. De fet, en analitzar adequadament la demanda de trànsit, és possible predir les condicions futures del trànsit, i utilitzar aqueixa informació per a l'optimització de les rutes preses pels vehicles. Aquest enfocament pot ser especialment efectiu si s'aplica en el context dels vehicles autònoms, que tenen un comportament més predictible, i això permet als administradors de la ciutat mitigar els efectes de la congestió, com és la contaminació, en millorar el flux de trànsit de manera totalment centralitzada. La validació d'aquest enfocament generalment requereix l'ús de simulacions que haurien de ser el més realistes possible. No obstant això, aconseguir alts graus de realisme pot ser complex quan els patrons de trànsit reals, definits a través d'una matriu d'Origen/Destinació (O-D) per als vehicles en una ciutat, són desconeguts, com ocorre la majoria de les vegades. Per tant, la primera contribució d'aquesta tesi és desenvolupar una heurística iterativa per a millorar el modelatge de la congestió de trànsit; a partir dels mesuraments de bucle d'inducció reals fetes per l'Ajuntament de València (Espanya), vam poder generar una matriu O-D per a la simulació de trànsit que s'assembla a la distribució de trànsit real. Si fóra possible caracteritzar l'estat del trànsit predient les condicions futures del trànsit per a optimitzar la ruta dels vehicles automatitzats, i si es pogueren prendre aquestes mesures per a mitigar de manera preventiva els efectes de la congestió amb els seus problemes relacionats, es podria millorar el flux de trànsit en general. Per tant, la segona contribució d'aquesta tesi és desenvolupar una Equació de Predicció de Trànsit per a caracteritzar el comportament en els diferents carrers de la ciutat en termes de temps de viatge respecte al volum de trànsit, i

aplicar una regressió logística a aqueixes dades per a predir les condicions futures del trànsit. La tercera i última contribució d'aquesta tesi apunta directament al nou paradigma de gestió de trànsit previst. Es tracta d'un servidor de rutes capaç de manejar tot el trànsit en una ciutat, i equilibrar els fluxos de trànsit tenint en compte les condicions de congestió del trànsit presents i futures. Per tant, realitzem un estudi de simulació amb dades reals de congestió de trànsit a la ciutat de València (Espanya), per a demostrar com es pot millorar el flux de trànsit en un dia típic mitjançant la solució proposada. Els resultats experimentals mostren que la nostra solució, combinada amb una actualització freqüent de les condicions del trànsit en el servidor de rutes, és capaç d'aconseguir millores substancials en termes de velocitat faig una mitjana i de temps de trajecte, tots dos indicadors d'un grau menor de congestió i d'una fluïdesa millor del trànsit.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Currently we are experiencing a constant growth of urban areas both in terms of automotive parks and population density, both of which are prone to cause critical issues both for citizens and city authorities, including environmental pollution, noise, accidents, and the increase of Carbon Dioxide (CO_2) emissions. In fact, these problems are closely associated with traffic congestion, resulting in an increased travel time of vehicles associated with unwanted delays and inefficient fuel use. For this thesis we selected the city of Valencia, whose metropolitan area is the third largest in Spain, with a population of 791 413 inhabitants. The city of Valencia provided detailed data on the flow of traffic on each street and avenue, for every hour during one year, based on inductive-loop traffic detectors scattered throughout the city.

Concerning solutions to reduce the amount of traffic, or to improve the traffic flow in a city, usual strategics have been proposed such as using public transportation, limiting the access of vehicles to downtown areas depending on their license plate number, semaphore timing regulation (traffic light synchronization), or deployment of on-site traffic agents. In addition, a detailed traffic analysis is often necessary to find ways to improve traffic flow. Traffic management solutions typically require the use of simulators able to capture in detail all the particular characteristics and dependencies associated with real-life traffic. These simulations depend on different factors such as vehicle speed, vehicle density, traffic flow, and the environment itself. So, we see the need to propose a procedure that allows creating scenarios based on realistic mobility models that are useful when simulating large-scale traffic in urban areas.

Among these novel techniques of handling traffic, centralized route manage-

ment emerges as a solution offering authorities full control of the traffic flow, thus allowing traffic optimization to reach maximum levels of effectiveness by deciding the route to be followed by each individual autonomous vehicle. By gaining *a priori* knowledge of the traffic congestion status, it becomes possible to optimize the route of new vehicles, especially in the case of automated vehicles, which can rely on a centralized trusted agent to indicate the most adequate route for each vehicle. When progressing towards these new traffic management paradigms, it becomes mandatory to gain full awareness of the behavior of the different street segments of a city in terms of how travel time can vary depending on the number of vehicles simultaneously traveling on a particular segment. Therefore, we analyze, model, and characterize how traffic becomes distributed throughout the city, gathering details about the number of vehicles traveling along the different street segments, as well as the travel times of vehicles entering a segment, along with the number of vehicles already in that segment, thereby extracting a relation between street occupation and delay.

In the last few years, we have seen how a novel mobility paradigm focused on automated vehicles has emerged, and it is steadily gaining momentum. As autonomous vehicles gradually become ubiquitous in coming years, they also present new opportunities to improve traffic by endowing traffic managers with more intelligent ways to regulate traffic when compared with the usual strategies. Notice that such a centralized administration approach is much more beneficial than relying on the vehicle itself to decide the best route based on knowledge of the traffic congestion status beforehand, as the latter approach is prone to cause intermittent congestion problems in the different streets of a city. Therefore, we perform an experimental study where we evaluate the effectiveness of our route server at mitigating traffic congestion to provide routes to all vehicles in the city, and minimize travel times by balancing traffic throughout the city.

1.1 Objectives

The main objective of this thesis is to propose and implement a centralized route manager for autonomous vehicles able to optimize the flow of traffic with a high effectiveness so as to reduce the travel time of vehicles. To achieve this, the proposed route manager determines the route of each particular vehicle based on traffic congestion predictions. This route server aims to achieve traffic balancing, and we relied on specialized tools for vehicular traffic simulation and control to validate it.

To achieve our main objective, it becomes necessary to accomplish several specific objectives, as presented below.

Heuristic iterative, Validation and Extensions:

- Analyse which route generation module matches the data of the induction loops.
- Obtain the travel time of the vehicles of the real and simulated data.
- Propose a heuristic that allows to adjust the number of vehicles to those detected in the city of Valencia in the designated period.
- Generate an O-D matrix for traffic that resembles the actual traffic distribution.
- Compare the results obtained against other existing mobility data.
- Determine what degree of congestion is expected if certain conditions cause additional traffic when driving in the city.
- Demonstrate how to regulate the overall number of vehicles in the city.
- Evaluate the impact of vehicle flow changes on the overall traffic congestion levels.

Modeling and Characterization of Traffic Flow:

- Propose a procedure to unify street segments when excessive fragmentation is detected if certain conditions are met.
- Predict the travel time associated to each segment for different degrees of congestion, being the latter measured as the number of vehicles located in the segment just before a new vehicle enters it.
- Determine the relationship between the number of vehicles in a segment, and the average travel time of vehicles for each particular segment.
- Perform regression to obtain the best curve fit describing the nonlinear relationship between segment congestion and travel time.
- Develop an equation to characterize travel times over a segment belonging to the sigmoid family.
- Apply logistic regression for improving the curve fitting results for most of the street segments under analysis.
- Perform a clustering analysis of the different street segments.
- Characterizing the different streets of the city in terms of vehicle load with respect to the travel time during rush hour traffic conditions.

Centralized route management solution:

- Link traffic and network simulators with our route server to offer a complete simulation environment through an intelligent communications interface.
- Propose a route server capable of handling all frequent updating of traffic in a city.
- Evaluate the effectiveness of our route server at mitigating traffic congestion.
- Balancing traffic flow by accounting for present and future traffic congestion conditions.
- Compare the routes taken by the vehicles within the reference traffic scenario against the scenario with a balanced load
- Demonstrate the improved traffic results to reduce congestion in some of the streets.
- Demonstrate the improvements obtained in terms of travel time and speed of the simulated vehicles.

1.2 Structure of the Thesis

The thesis follows a methodology that covers theoretical topics which are validated using simulators. On the one hand, we rely on a heuristic that allows creating scenarios based on realistic mobility models that are useful when simulating large-scale traffic in urban areas so as to improve traffic congestion modeling. Thus, we propose an equation to characterize travel times over a segment belonging to the sigmoid family; specifically, we apply logistic regression, being able to significantly improve the curve fitting results for most of the street segments under analysis. Finally, we introduce a novel traffic management paradigm by proposing a route server capable of handling all the traffic in a city, and balancing traffic flows by accounting for present and future traffic congestion conditions. Thus, we perform a simulation study using real data of traffic congestion in the city of Valencia, Spain, to demonstrate how the traffic flow in a typical day can be improved using our proposed solution.

This dissertation is organized in 9 chapters. Below, we briefly describe the contents of each part:

- **Chapter 2. Vehicle network simulators**: We provide general aspects and features about vehicle network simulators.
- **Chapter 3. Combining ABATIS, OMNeT++ and SUMO simulation tools**: We introduce the tools that have been used for the development of this thesis.

- **Chapter 4. Traffic flow generation methodology**: We describe the procedure that has been followed to obtain an Origin-Destination matrix for the city of Valencia.
- **Chapter 5. Traffic congestion analysis**: We study the impact of varying the traffic load for the city of Valencia, and determine what degree of congestion can be expected if certain conditions cause additional traffic to circulate in the city, and thus have a proper traffic planning and optimization.
- **Chapter 6. Traffic characterization and modeling procedure**: We describe the individual characterization of street segments in terms of average travel times experienced by vehicles for different degrees of congestion based on the number of vehicles that are ahead of a vehicle when entering a segment.
- **Chapter 7. Traffic congestion behavior**: We describe the adopted equation based on the logistic family for the regression analysis that better adopts the behavior of street segments in the city of Valencia.
- **Chapter 8. Proposed traffic balancing scheme**: We describe the procedure followed to improve the flow of traffic in the city of Valencia, focusing on achieving the load balancing of vehicles.
- **Chapter 9. Conclusions, publications and future work**: We conclude the thesis with the results obtained, together with the published publications and future works.

At the end we include a list of acronyms and the bibliography.

Chapter 2

Vehicle network simulators

2.1 Introduction

Concerning vehicular network simulators, there are several tools available to carry out simulations with different characteristics, for which they have a significant dependence on integration with other programs to improve their accuracy. Currently, these tools have great acceptance in the scientific community, and they are classified into two categories: traffic simulators, and network simulators, as we can see in Figure 2.1.

Next, the existing traffic models that have the required level of detail to perform the simulations will be presented. Thus, the mobility models are representations of the elements of the scenario to be simulated, and their primary objective is to imitate the behavior of the various elements in the network simulation, for which aspects related to the behavior of the drivers, the traffic signals, acceleration and speed changes associated with vehicles, etc., are observed.

2.1.1 Vehicular traffic models

Traffic management solutions typically require the use of simulators capable of capturing in detail all the particular characteristics and dependencies associated with real-life traffic. The simulations depend on factors such as the speed of vehicles, the vehicular density, the simulation environment, and the traffic flows. Vehicle traffic models can be classified according to the level of detail to represent the traffic system as four types of models: Macroscopic, mesoscopic, microscopic, and sub-microscopic [1]. Figure 2.2 shows the three main vehicle traffic models adopted for simulation.

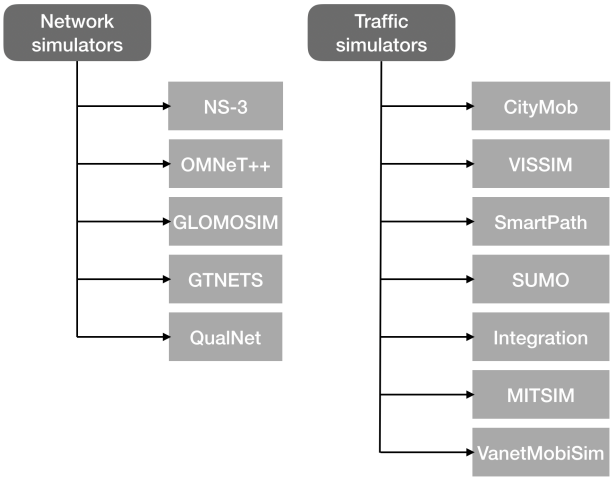


Figure 2.1: Simulators by category.

Macroscopic models simplify the analysis by describing traffic at higher aggregation levels. This model assumes the traffic is properly located in the lanes of the roads, and the maneuvers of each vehicle are not represented explicitly. Although their computational requirements are low, their results are not too representative or accurate.

Mesoscopic models represent the transportation system through the analysis of driver behavior. This model does not distinguish the individual traces of the vehicles, but rather the individual behavior specifically, based on the probability functions required for a vehicle in motion with a certain defined speed, as well as the probability being located at a particular position, at a specific time. Traffic is represented by small groups of traffic entities, iterations and activities, thus introducing a moderate level of detail. An instantaneous event, such as the maneuvering lane change, is represented by an individual vehicle, where the decision to perform this action is based on different speeds and relative lane densities, resulting in a high computational cost.

Microscopic models describe the time-space traffic behavior so as to include vehicles and drivers, as well as their interaction, in an individual manner, thus reaching a high level of detail. Therefore, microscopic models are able to represent real situations with a high accuracy since each vehicle becomes an independent node in the simulation.

Finally, sub-microscopic models describe the individual characteristics of the vehicles in the traffic, detailing the behavior of the driver, the control of the vehicle (such as changing gears), and intelligent cruise control operations, among others, in correspondence with several conditions that are modeled in detail. In addition,

the operations of sub-units, or specific parts of the vehicles, are described, which increases the computational complexity.

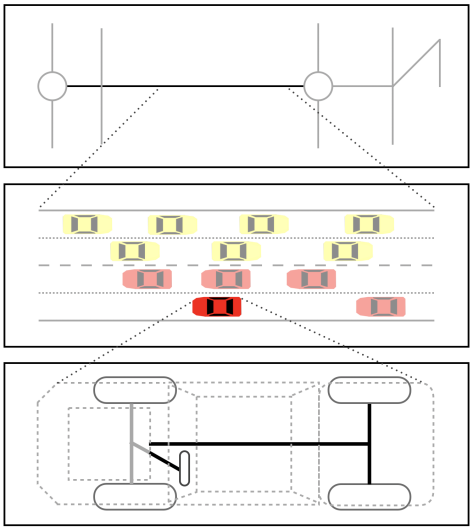


Figure 2.2: Vehicular Models (from above macroscopic, microscopic, sub-microscopic).

2.1.2 Mobility models

In order to realistically simulate vehicular networks, it is necessary to use a mobility model in which the topology of the map, the nodes, and the movements of the vehicles, including speed, direction and acceleration, can be represented. As we can see in Figure 2.3, these models can be classified into synthetic models, survey-based models, trace-based models, and traffic simulator-based models [2].

Synthetic models

A mathematical model such as stochastic, traffic stream, car-following, and queue models, is developed based on driver behavior. These are modeled to simulate the various actions of vehicles in a virtualized manner.

The problem of these models is the complexity to model human behavior, since they cannot be programmed to follow a specific behavior that contemplates all events.

A model called Obstacle mobility [3] defines movements for the vehicles so as to resemble a real environment, thus allowing the user to define a scenario based on the dimensions of the area, and add certain obstacles such as other

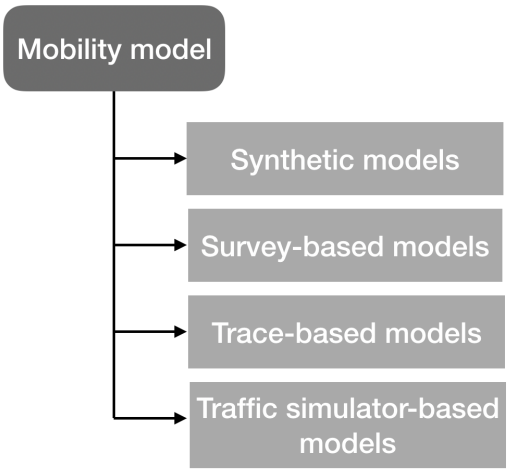


Figure 2.3: Classification of mobility models.

vehicles, buildings, etc. To perform the calculations of the potential routes it is necessary to provide the obstacles previously, using a simple concept based on the minimum distance to reach a reference point called Voronoi Diagram of the obstacle vertex, and determine the routes that are in the middle of two adjacent obstacles. Then, the vehicles begin to be located by selecting both a destination and a speed randomly, and following the path of minimum distance from the Voronoi diagram where the cost of each segment of the route is its own Euclidean distance. Once it reaches its destination, the vehicle stops for a predetermined period, selects a new point as a destination, and calculates the new route.

Survey-based models

These models extract the mobility patterns of surveys that contain information about the urban circulation obtained from previous studies, and develop a generic mobility pattern by deterministically reproducing the patterns corresponding to the real urban traffic. Usually, these models are applied to macroscopic traffic models.

The main issue with this model is to build an infrastructure able to achieve the abstraction and accurate monitoring of human behavior so as to generate a generic model based on the data collected.

Thus, there are examples of realistic vehicular traces [4] that obtain traces of a microscopic traffic simulator called Multi-agent Microscopic Traffic Simulator (MMTS), developed at the Technical University of Berlin in Germany, which is able to simulate real public and private traffic of Switzerland through real regional

road maps with a high level of realism.

Trace-based models

These models generate patterns from mobility traces and extracting generic patterns from surveys, without involving mathematical equations. Therefore, these models are used for monitoring the movement of vehicles and pedestrians, obtaining the general traces according to their location and movement.

The difficulty of these models is to extrapolate the patterns not directly observed by these traces, and for which complex mathematical models are created that allow predicting, to a certain degree, the vehicular mobility patterns based on traces, since vehicle flow details are not available.

As a good reference to this model is Reality Mining [5], whose main objective is to study the characteristics of mobile devices in order to compare human interactions in greater depth in comparison with the methodology based on surveys or simulations. The reason why it was not done before, is because the mobile phones were not powerful enough to be able to track their users.

Traffic simulator-based models

These models are based on an intense validation of processes in real traces or behavioral studies. The characteristics that are highlighted for this type of model are urban microscopic traffic, energy consumption, and the monitoring of noise and pollution levels. Traffic simulators need interfaces to connect to network simulators to analyze the traffic along with the network simulator input files. Therefore, users can validate simulated traffic patterns with a high level of granularity. So, they are an evolution of their predecessors, and the most widely used nowadays.

The drawback of this model is the complex configuration of traffic simulators, due to the calibration required by a wide set of parameters. In addition, the level of detail required by a vehicular network simulator is relatively low compared to the traffic analysis made by other simulators.

For this model, it is represented by Simulation of Urban MObility (SUMO) [6], which is responsible for managing the real vehicular mobility based on synthetic or realistic traces. In addition, for each step of time, the speed of the vehicle is adapted to the speed of the vehicle ahead to avoid collisions following the Gipps model [7] with its Equation 2.1 that offers a safe speed where b is deceleration, Δt is time of the driver's reaction, s is the distance to the vehicle ahead, and v_1 is the speed of the vehicle ahead.

$$v_{safe}(s, v_1) = -b\Delta t + \sqrt{b^2\Delta t^2 + v_1^2 + 2b(s - s_0)} \tag{2.1}$$

2.2 Traffic simulators

Due to the increased power of computer technologies and the invention of Intelligent Transport Systems (ITS), traffic simulation has become one of the most widely used approaches to the analysis of traffic in the design and evolution of traffic systems. The ability of traffic simulation to emulate the temporal variability of the phenomena that occur in traffic, make it a unique tool to capture the complexity of traffic systems.

In recent years, traffic simulation has more scientific and professional acceptance, so there is a wide variety of traffic simulators used by thousands of users, researchers and public agencies. These simulators are based on vehicle traffic, and offer the possibility of modeling several real scenarios without forgetting the characteristics of the elements that compose this environment, such as the characteristics of vehicles and drivers, trains, pedestrians, etc., that are interacting with the elements of the routes such as traffic lights, obstacles, crossings, parking, etc.

Also, the movement of vehicles in the roads with certain scenarios, and their respective mobility model, can be configured in advance and includes factors such as speed, acceleration and time. In addition, they allow mobility traces to be generated which in many cases can be used by network simulators.

In addition, these simulators have the ability to import real maps from different sources such as OpenStreetMap (OSM) [8], Navigation Technologies Corporation (NavTeq) [9], Verkehr Im SIMulationsmodell (VISUM) [10], Multi-Agent Transport Simulation (MATSim) [11], Verkehr In Städten - SIMulationsmodell (VISSIM) [12], OpenDRIVE [13], and ArcView [14]. In this work, the maps used for our studies will be imported from the OSM using the SUMO traffic simulator.

2.3 Network simulators

To evaluate the performance of an ITS service or protocol, not only mobility models are necessary, but also the complete protocol stack for communication between the elements that compose an ITS are required. In order to evaluate the performance of any network and observe any problem that may exist, it is necessary to use network simulators.

Currently, there are several simulators to test and evaluate the new protocols and the different applications for ITS. These simulators are characterized by modeling the interconnection between machines, routers and other devices. In addition, they offer the possibility of performing multiple types of tests and obtaining generally reliable results, without needing to deploy a real network or to intervene in an active network.

Some network simulators allow mobility patterns generated by traffic simulators, including Application Programming Interfaces (APIs) in order to establish

a bidirectional communication between them that increases the reliability and accuracy of the simulations.

The Objective Modular Network Testbed in C++ (OMNeT++) [15] simulator will be used in this work, which has an framework called INET [16] allowing communication between the SUMO traffic simulator and the OMNeT++ network simulator.

Chapter 3

Combining ABATIS, OMNeT++ and SUMO simulation tools

In this chapter, we detail the network simulator and the traffic simulator used in this thesis, called OMNeT++ and SUMO, respectively. Both simulators are open source, and they can communicate with each other through a communication module that works as an extension of SUMO. In addition, SUMO has additional tools that can greatly enhance this simulator. Another point that will be detailed is the route server called Automatic Balancing of Traffic through the Integration of Smartphones with vehicles (ABATIS) [17], and how is connects to OMNeT++, so that combined they provide a full simulation framework that allows performing any sort of study focused on traffic management in a city.

3.1 SUMO Traffic Simulator

3.1.1 Overview

Most people have the idea that traffic characterization would merely rely on a few variables, such as the departure time, the route that the different vehicles will follow, and in some cases the duration of a route. However, this assumption is not correct, as it will obviously depend on the vehicle, the driver, and the state of traffic congestion along the route, representing several problems in modeling traffic, so that the description of traffic is achieved using complex mathematical equations that measure these two factors: (a) the movement of the vehicle on one way; and (b) the human behavior, both of which are correlated with each

other. Other parameters that influence traffic are the weather conditions, the infrastructure, and the incidences of the roads.

In this area, it is worth pointing out the Institute of Transportation Systems at the German Aerospace Center (DLR), which was able to obtain important conclusions in the scope of a traffic management systems through a project developed by different agencies and co-funded by the European Commission to develop micro-simulation tools to help solve road traffic through the Simulator Modelling Appliyed to Road Transportation European Schema Test (SMARTEST) [18] project. In this project, the DLR proposes to develop an open source tool capable of satisfying the needs of microscopic and continuous road traffic simulation, with all its parameters so much for pedestrians, vehicles, traffic lights, environmental zones, infrastructures, etc.

Thus, SUMO emerges as an open source package for traffic simulation that is constantly evolving, and that has a complete range of traffic modeling utilities, including the ability to import road networks information in multiple formats, to estimate demand, and providing routing utilities from various sources of input as Origin-Destination (O-D) matrices, traffic counting, etc. In addition, it offers a high portability and high performance in simulation through the Traffic Control Interface (TraCI) [19] communication allowing online simulations.

3.1.2 Basic paradigms and characteristics

SUMO has the potential to simulate traffic in a road network of a city with the advantage that its simulation can be multimodal, including not only the movement of vehicles within the city, but also aspects such as the public transport system, train networks, bicycle networks, and even pedestrian routes. This advantage of the simulator can be described through multiple routes, which can be composed in sub-routes, and thus be able to describe a single traffic mode.

By simulating a person, simulator can take a vehicle to a nearby public transportation station and continue the trip, as well as walk or take a public bicycle. The process of walking inside the simulation is not stipulated, so an estimated modeling of the time that the person needs to find their destination is performed. An example summary of this type of routes in a multimodal simulation is shown in Figure 3.1.

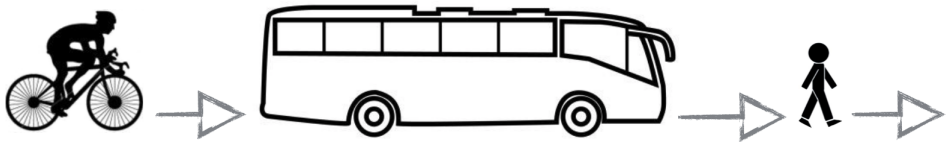


Figure 3.1: Multimodal traffic Simulation.

The flow of traffic is simulated microscopically meaning that, within the simulated network, each vehicle is located in a certain place with relevant information such as time of departure of the vehicle, its speed, and the streets where it will pass. Each time lapse in the simulation causing parameter updating has a duration of one second. In this way, a continuous movement in space can be simulated discretely. The traffic simulation must respect the speed and priority rules towards the right side, which are street attributes.

Although SUMO was not originally designed to address autonomous vehicles, it is a very flexible tool which can include different driver models. In fact, SUMO has been recently updated so as to model autonomous vehicles, and there are several scientific publications that use this mobility simulator for that purpose [20][21]. For this work, the simulator has been used in its version 0.32.0 since it is open source, it is in constant development by collaborators and scientists, having a good acceptance in the scientific community. For this reason, it has functionalities that enhance its efficiency when performing a traffic simulation such as:

- Microscopic simulation of vehicles, pedestrians and public transport modeled explicitly.
- Online interaction using communication interfaces to control the simulation through TraCI.
- Multimodal simulation of traffic such as vehicles, pedestrians, and public transport.
- Schedules of traffic lights that can be imported or generated automatically by SUMO.
- Support formats to import maps of external agents such as OSM, NavTeq, VISSIM, VISUM, OpenDrive, MATSim, and ArcView.
- Only uses portable libraries and is mainly implemented in the C ++ programming language.

3.1.3 Additional tools

The SUMO traffic simulator consists of more than one application to allow us to process input data necessary for the simulations, with their respective parameters. Below we present the most important tools of this simulator.

NETCONVERT [22]

Due to its high complexity, the description of the network that SUMO uses can not be generated by a user, being necessary the use of a tool to convert this information

on critical elements, such as the list or codes of the streets that SUMO handles named "*Edges*", and optional nodes to complete the network that SUMO requires.

During the process of converting the format of an external agent to the readable format for SUMO, this tool reads the data, computes the inputs you need for the simulator, and writes the results in eXtendible Markup Language (XML) files. Usually the Netconvert input data are:

- SUMO native files: *.edg.xml, *.nod.xml, *.con.xml, *.tll.xml, *.add.xml
- VISUM adn VISSIM: includes forms and demands of their networks
- OSM: *.osm.xml
- OpenDrive: *.xodr
- MATSim: *.xml
- ArcView: *.shp, *shx, *.dbf, *.proj
- NavTeq: extracts from the NavTeq GDF database, that is converted using specialized tools to read unsplitted network versions.

For the specific case of the city of Valencia, Spain, a map with the OSM format has been imported into the network format of SUMO traffic simulator.

In Figure 3.2 we can see how data can be computed from a simple list of nodes and edges, besides adding traffic flows from left to right, and from top to bottom. The first step is to determine the different priorities in the unions; the second step is to perform a computation of relations between the lanes and the edges that can be raised, and the last step is to divide the destinations between the incoming lanes. Depending mainly on the number of incoming and outgoing edges, their respective sizes, and the priorities within the network, the calculation is flexible.

But nevertheless, there is a lot of information within the map of any city, as routes for bicycles, trains, boats, emergency services, pedestrians, etc., that are of no use to us for our study. Thus, we proceed to exclude those extra information for obtaining a map corresponding only to vehicular routes. For this propose, we used the following command:

netconvert -osm valencia.osm.xml -remove-edge.by-vclass rail_slow, rai_fast, bicyle, pedestrian, lightrail, cityrail, delivery -output-file valencia.net.xml

DUAROUTER [23]

DUAROUTER is a tool that imports different definitions of demand presented by the vehicles, and then computes the vehicular routes that pass through those edges, and uses those routes as the shortest ones.

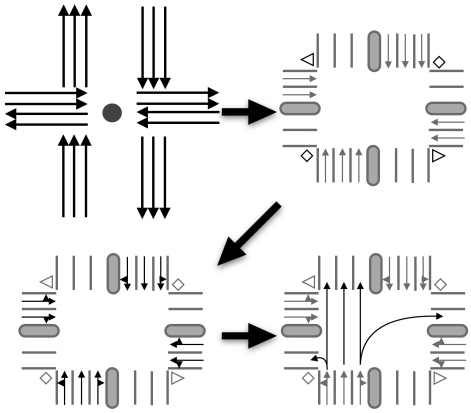


Figure 3.2: Flat network conversion to a complete description.

The purposes of DUAROUTER are to build routes for vehicles according to the corresponding demand definitions, computing routes during the user's assignment, and repairing the different connectivity problems existing in the route files.

The routing algorithms that this tool use are:

- Dijkstra: determines the shortest path given a vertex of origin to the rest of vertices in the graph with a weight in each edge that depends on time.
- A*: determines the lowest cost path between a source node and an objective node as long as it accomplishes certain time-dependent conditions.
- Contraction Hierarchies (CH): This algorithm is preprocessing-based, being efficient at the time of having a large volume of queries, because it considers time-dependent weights. The preprocessing is done without restrictions in the vehicle class, and reduces its efficiency in multimodal networks.
- CH Wrapper: being a variant of the CH, this algorithm performs a separate preprocessing for each vehicle class that is found, increasing the routing efficiency.

OD2TRIPS [24]

This tool performs calculations of travel tables called O-D trips in the form of a matrix. To achieve this, it is assumed that the matrices should be coded according to the volume of vehicles from a district or Traffic Assigment Zone (TAZ) to another area.

Usually, this tool can only import data stored in the VISUM and VISSIM formats. In case an user does not have matrices stored in these formats, there are three possibilities to import them:

1. Convert them to a supported format defined by the user.

2. Write your own reader of OD2TRIPS.

3. Convert them to flow definitions, and pass them to DUAROUTER.

The OD2TRIPS tool reads all the matrices and generates trip definitions, and they are numbered from zero. Each cell of the O-D matrix describes the number of vehicles that will be inserted into the network during a certain period, and calculates the times of departure of the vehicles. This is usually done randomly within a designated interval, describing the cells and uniformly allocating the time between intersections.

ACTIVITYGEN [25]

This tool allows generating demand associated with the description of the population in the network, using a traffic model based on activities of the population such as work, school or vacations, as well as for pedestrians while walking, and for transportation including boats, vehicles, bicycles and buses. In other words, the main purpose of ACTIVITYGEN is to generate demand for a synthetic population, and estimate mobility with respect to the behavior of that population.

This tool uses a statistical file, which includes general information about the city such as the number of inhabitants, the rate of employment and unemployment, the number of homes, incoming and outgoing traffic, etc., as we can see in Figure 3.3.

JTRROUTER [26]

This package uses definitions of junction turning ratios and traffic volumes to calculate the possible routes that would exist in the vehicular network. This approach can be used to create demand within a part of the road network in a city that contains up to ten nodes. The input parameters for this tool are:

- The description of the vehicular flows of the scenario.

- The network to route the vehicles.

- Description of the turning ratios for the junctions in each interval, or description for each edge an usage turn probability.

```
<city>
    <general inhabitants="12300" households="6340" childrenAgeLimit="18" retirementAgeLimit="65" carRate="0.9" unemploymentRate="0.02"
        footDistanceLimit="450" incomingTraffic="2500" outgoingTraffic="3000" />
    <parameters carPreference="0.5" meanTimePerKmInCity="360" freeTimeActivityRate="0.15" uniformRandomTraffic="0.2"
        departureVariation="120" />
    <population>
        <bracket beginAge="0" endAge="18" peopleNbr="1400" />
        <bracket beginAge="18" endAge="75" peopleNbr="2990" />
    </population>
    <workHours>
        <opening hour="30600" proportion="0.3" />
        <opening hour="32400" proportion="0.7" />
        <closing hour="45000" proportion="0.2" />
        <closing hour="63000" proportion="0.2" />
        <closing hour="64800" proportion="0.6" />
    </workHours>
    <streets>
        <street edge="10305544#0" population="7.95" workPosition="1.59"/>
        <street edge="10305544#1" population="6.7" workPosition="1.34"/>
        <street edge="10305544#2" population="8.9" workPosition="1.78"/>
        ....
        ....
        ....
        <street edge="8049792" population="21.4" workPosition="4.28"/>
        <street edge="8049793" population="26.45" workPosition="5.29"/>
        <street edge="8049794" population="9.5" workPosition="1.9"/>
    </streets>
</city>
```

Figure 3.3: Statistical file of the ActivityGen tool.

DFROUTER [27]

Among the packages included in the traffic simulation SUMO version 0.32.0, this routing module has been designed for road scenarios based on the idea that the roads are equipped with induction loops, measuring the inflow and outflow of the roads. This tool allows to reconstruct the number of vehicles and routes in the simulation of the vehicular network, the information of type of vehicles, flow and speed to achieve the desired O-D matrix.

The particularity of this tool is to start from induction loops counts for all the different streets of a scenario or city, estimating the current vehicular routes that match such input. The algorithm used by DFROUTER works if the network is partially or completely covered with induction loops, and generates paths by accounting for all possible O-D pairs [28].

Figure 3.4 shows the different elements associated with the process adopted by the DFROUTER tool. Thus, it requires flow input data from the road detectors, and also the road network of the scenario to be simulated. Combining them, DFROUTER will provide the output data of the routes, as well as the origin and destination of the different vehicles, including relevant details such as departure speed, departure time, departure lane, and departure position in the edge.

The steps that this tool follows are:

1. Import the vehicular network, including the position of the detectors with their respective measurements.

2. Apply the classification of the detectors in the following categories: i) source detectors are those initial points of the routes, ii) in-between detectors are

those located in the middle of at least two detectors, iii) sink detectors are usually the endpoints of the routes.

3. Calculate the vehicle flow between consecutive detectors.

4. Compute the routes using probabilities. Normally, measurements are provided on a per-lane basis, and they need to be summarized for each cross-section.

The main requirement of the DFROUTER tool is that the road network must contain at least one induction loop in the main road segments; secondary streets, without the induction loops, are nevertheless taken into consideration for the route calculation. Other sources of traffic flow information that differ from the standard induction loops are not supported. In other words, DFROUTER requires a network to contain a list of the induction loop detectors that include their positions and the associated vehicle count.

An induction loop is considered a source detector when there is no other detector in the same street or in any nearby or foregoing street, meaning that the vehicles are assumed to start their routes from those points. In-between detectors are the most common within cities, being required when our objective is to maximize the match between the data from the reference sensors and the simulated measurements. In addition, these detectors can act as starting points for the vehicles. Finally, sink detectors are used when there is no other detector on preceding streets or in streets that follow; unless strategic parts of the city are clearly known to act as traffic sinks, this type of detectors can be discarded. for

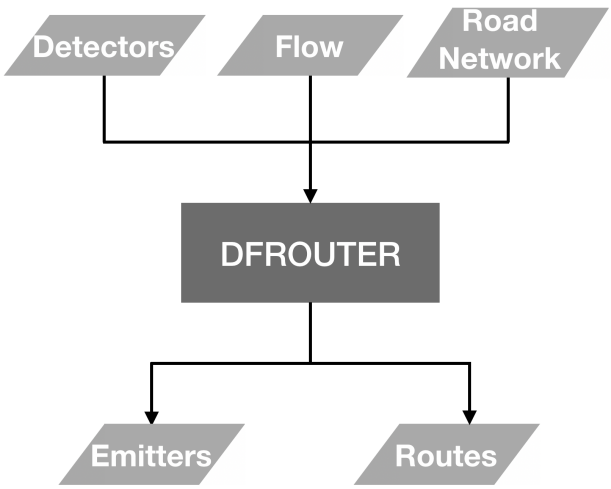


Figure 3.4: Functional flowchart of DFROUTER.

adequate processing, all these data should be in a file with an XML extension, similar to the one shown in Figure 3.5

The route data generated by DFROUTER is simply based on flow data for the target edges. This method works similarly to a static O-D matrix estimation, but it does not work as an O-D estimator and can not be compared to O-D estimation methods such as maximum entropy, generalized least square, Bayesian inference, minimum information, etc. [28].

However, there are methods that can be compared with the DFROUTER algorithm, including: (a) an equally divided O-D matrix that assigns all destinations in the same portion, but with implausible results; (b) proportional O-D matrix, where the attraction of any destination is a function of the number of trips completed in that destination, as this method is identical to the one adopted in DFROUTER; (c) iterative methods which balance input and output flows through an iterative tuning algorithm, being identical to that used in DFROUTER; (d) gravity models which manage a concept that the probability of very long and very short trips on motorways is low, while in DFROUTER it depends heavily on the treatment of external stations; and (e) turning percentage is an identical method to DFROUTER.

Once the information concerning the points of entry and exit of the vehicles in the network has been obtained, a third step is to determine the routes for each of the origins and destinations. For this purpose, a file is generated detailing all the routes considered, but with no information regarding the number of vehicles associated with each route. Commonly, the routes start at the source detectors and end at the corresponding sink detectors. However, there is an additional option for this tool to force the start of the routes built in the middle of the detectors. Thus, we can see in Figure 3.6 an example of the routes file.

Subsequently, DFROUTER combines the calculated routes with the flow information taken from the actual sensors of induction loops to determine the number of vehicles associated with each route. Fulfilling this task requires a file that contains the following information: identification of the detector, initial time, number of vehicles driven through the detector, and the average speed for those vehicles, as illustrated in Figure 3.7.

In the last step, DFROUTER generates the real O-D matrix allowing it to

```
<a>
    <detectorDefinition id="dd_22740816#2_0" lane="22740816#2_0"   pos="1.0"/>
    <detectorDefinition id="dd_22740816#3_0" lane="22740816#3_0"   pos="1.0"/>
    <detectorDefinition id="dd_22740817_0" lane="22740817_0"   pos="1.0"/>
    ....
    ....
    <detectorDefinition id="dd_22740818_0" lane="22740818_0"   pos="1.0"/>
    <detectorDefinition id="dd_99867904_0" lane="99867904_0"   pos="3.0"/>
</a>
```

Figure 3.5: Induction loop detectors.

```
<routes>
    <route id="4693573#1_to_98711574#1" edges="4693573#1 4693573#2 4693573#3 98711556 98711574#0 98711574#1"/>
    <route id="4693573#1_to_98711543#2" edges="4693573#1 4693573#2 6277183#0 98711543#1 98711543#2"/>
    ...
    ...
    <route id="4693573#1_to_6277192#3" edges="4693573#1 4693573#2 4693573#3 98711556 6277192#0 6277192#1
        6277192#2 6277192#3"/>
</routes>
```

Figure 3.6: Route data file generated by DFROUTER.

store information about the total number of vehicles along with its routes. With respect to the vehicles, their routes start at any source detector located along the associated routes, and those vehicles will have their information in another file that is consistent with the flow file, and previously computed routes as can be seen in Figure 3.8.

The DFROUTER tool to be configured can include the input files (flow, detectors and road networks) and generate the respective output as a routes file, and the vehicles with their respective information in another file, as shown in Figure 3.9 for the city of Valencia.

In this way, we present in Figure 3.10 a summary of the possibilities for configuring demand for traffic simulation using the tools that are included in the SUMO traffic simulator.

```
Detector;Time;qPKW;qLKW;vPKW;vLKW
dd_98711537#6_0;0;102;0;20.002;0
dd_98711537#6_1;0;103;0;20.002;0
dd_98711543#2_0;0;102;0;20.002;0
dd_98711543#2_1;0;103;0;20.002;0
```

Where:

$Detector$:Identifier of the detector. They must be equal to the induction loop detectors file.
$Time$:Time period begin the entry description in minutes.
$qPKW$:Number of vehicles that drive over the detectors.
$qLKW$:Number of public vehicles that drive over the detectors.
$vPKW$:Average speed of the vehicles that pass over the detectors in Km/h.
$vLKW$:Average speed of public vehicles passing over the detectors in Km/h.

Figure 3.7: Flow information file.

```
<additional>
    <routeDistribution id="dd_4693573#1_0">
        <route refId="4693573#1_to_987115744#1" probability="0.07"/>
    </routeDistribution>
    <vehicle id="emitter_dd_4693573#1_0_0" depart="0.00" departLane="best" departPos="base"
        departSpeed="max" route="4693573#1_to_9281469#0"/>
</additional>
```

Figure 3.8: Vehicle information file generated by DFROUTER.

3.2 The OMNeT++ network simulator

3.2.1 Overview

OMNeT++ is an extensible and modular framework that includes a simulation library with components based on the C++ object-oriented programming language. This network simulator is primarily aimed at building network simulators with a generic architecture. Thus, it is able to perform simulations of events, and solve several problems, as presented below [29]:

- Modeling of network communications in both wired and wireless environments.
- Modeling of Internet protocols used in the network.
- Modeling queueing networks.
- Modeling photonic networks.
- Modeling sensor networks

```
<configuration>
    <input>
        <net-file value="valencia.net.xml"/>
        <detector-files value="valencia.det.valenciaATA.xml"/>
        <measure-files value="valencia.flow.valenciaATA.txt"/>
    </input>

    <output>
        <routes-output value="valencia.rou.valenciaATA.xml"/>
        <emitters-output value="valencia.add.valenciaATA.xml"/>
    </output>
    ...
    ...
</configuration>
```

Figure 3.9: DFROUTER configuration file.

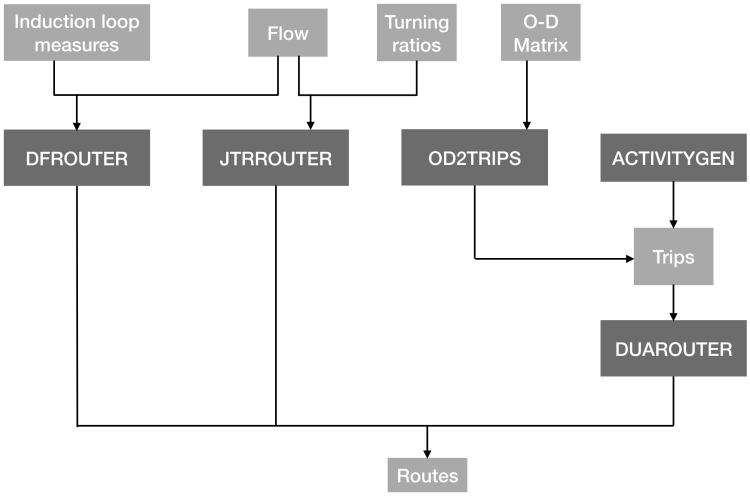


Figure 3.10: Supported tools in SUMO for route generation.

- Modeling wireless ad-hoc networks.
- Multiprocessor modeling and other distributions of hardware systems.
- Validation and modeling of hardware architectures.
- Modeling of on-chip networks.
- Performance evaluation in complex software systems.
- Modeling and simulation of any system based on events that can be mapped in entities that exchange messages.

This simulator also provides modules or writer components in C++, as well as a high level language called NEtwork Description (NED) for the use of these models. NED is the description language of the OMNeT++ topology, since it provides a simple but powerful syntax when it comes to defining regular topologies. One of the advantages that can be taken into consideration is the re-usability of the models since they are open source. The emergence of this network simulator dates from 1997 in the Telecommunications Department of the University of Technology and Economics of Budapest, Hungary [30].

3.2.2 Modular structure

The models used by OMNeT++ are mainly composed of hierarchical models, which are divided into basic or simple modules that can be regrouped in other

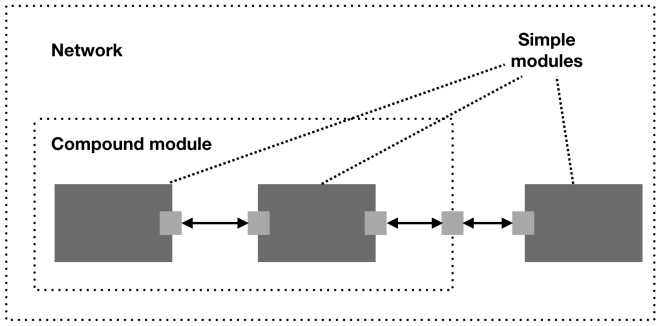


Figure 3.11: Simple and compound modules of OMNeT++.

composite models, and all of them are within a higher level called the system module, as shown in the Figure 3.11.

Among the basic modules there is one called type, which offers the basic functionality to implement simple and compound models. When creating the modules, the user defines their type and the instances of these types of modules are used as components for more complex modules. Finally, user creates the system module with an instance of a previously defined module type. All the modules of the network are instantiated as sub-modules. This allows the user to transparently divide a single module into several simple modules within a compound module.

For communication, messages with traditional attributes such as timestamp and arbitrary data are used. Typically, the sending of messages between simple modules is done through gates. These gates are inputs and outputs of the interfaces of the modules, so the messages are sent to the output of the gates and entered by other gates of entry. These inputs and outputs are known as connection. In the hierarchical structure of the module, the messages travel through a chain of connections that start and finish in a simple module. Likewise, compound modules are like a module each one that transparently exchanges messages both internally and externally.

The parameters that the modules can have, are used to pass the configuration data to the simple modules, and in this way to help define the topology module either by a file with NED extension or a configuration file (omnetpp.ini). Parameters can be of various types, such as strings, numerical values, and pointers. Because the parameters are represented as objects in the program. Compound modules can pass parameters or expressions to their sub-modules through values or references. Those values can be propagated globally in the module, and thus be changed during the execution of the simulation. This technique is useful in the case of the simulation of scenarios for the optimization of parameters. For our specific case, Table 3.1 represents the most relevant simulation parameters.

Table 3.1: Simulation parameters for the city of Valencia.

Parameter	Values	Description
sim-time-limit	900s	Simulation time limit
manager.updateInterval	1s	Simulation update interval
manager.host	localhost	Machine where the server is running
manager.port	10100	Communication port
manager.moduleType	valencia.Car	Base module for simulation
manager.moduleName	host	Module name
manager.lauchConfig	xmldoc('lauch.sumo.xml')	File name for simulation with SUMO
manager.pathLinkABATIS	../linkABATIS.py	Interface for connection with ABATIS
manager.pathAdditionalfile	../valencia.add.xml	Emitters file path of Valencia
manager.pathDataBase	../valencia.DB	Database path of Valencia.
manager.pathOSM	../valencia.osm	OSM file path of Valencia
manager.pathRoutesFile	../valencia.rou.xml	Route file path of Valencia
manager.additionalStatistics	true	Additional statistics activate/deactivate.
manager.pathStadisticalFile	../VehicleArrived.sta.csv	Statistics files of vehicles arrived
manager.pathVehicleNoArrived	../VehicleNoArrived.sta.csv	Statistics files of vehicles no arrived
manager.pathLearningFile	../VehicleLearning.sta.csv	Segment learning file path

3.2.3 Simulation paradigms

The OMNeT++ network simulator contains a large set of libraries to perform simulations because it must meet the needs of the tasks of a simulation. In addition, it offers the developers routines to have control and with a graphical interface for the creation of models, as well as the execution and debugging of the simulations. Thus, an additional advantage is the ability to support the traffic in the network, since it provides the ability to explore the current network topology, and from which a graphical data structure is extracted.

This simulator allows the user to define the structure of the modules to be simulated through the NED language. The fundamentals of a NED description are the declarations of the simple modules, the definitions of the compound modules, and the definitions of the network to be used. In particular, the simple modules describe the module's interface, the compound modules focus on the declaration of the interfaces of the external models, the submodules and interconnections of the modules, and finally the definition of the network defines an instance of a type of module.

Normally a module consists of the following parts:

- Messages of definitions which users can define various types of messages, and these in turn can add the data that is produced. In addition, these messages

will be translated into the C++ language.

- Simple modules are written in the C++ programming language with their respective .h and .cc suffixes.

- The topology of the NED language description are described by a modular structure with its corresponding parameters. The advantage of using OMNeT++ to open these types of files is to offer both a graphical environment and a text editor at the same time.

3.2.4 INET Framework

INET is an open source framework for OMNeT++ simulation environments. This framework provides protocols, agents, and models for the work of researchers in areas such as communication networks. Thereby, INET is mostly used when new protocols are designed and validated, or new scenarios are explored.

INET supports an extensive class of communication networks that includes wired, mobile, wireless, ad hoc and sensor networks. So, this framework contains implementations for the most relevant Internet protocols such as Transport Control Protocol (TCP), User Datagram Protocol (UDP), Internet Protocol version 4 (IPv4), Internet Protocol version 6 (IPv6), Open Shortest Path First (OSPF), Border Gateway Protocol (BGP), etc. In addition, it offers implementations for different protocols in the lower layers (physical and link level) such as IEEE 802.11, Ethernet, Point-to-Point Protocol (PPP), various Medium Access Control (MAC) sensor protocols, etc. It also includes protocols that support mobility, such as routing protocols in Mobile Ad hoc NETwork (MANET), advanced Quality of Service (QoS) functionalities such as MultiProtocol Label Switching (MPLS) with Label Distribution Protocol (LDP), Differentiated Services (DiffServ) and Resource Reservation Protocol - Traffic Engineering (PSVP-TE), some application models, and other components and protocols.

As part of this framework is TraCI [19], which is an open source architecture based on a TCP connection between OMNeT++, which acts as a client, and the SUMO traffic simulator, that acts as a server. Among them, the communication is through messages in a bidirectional way to exchange their commands. Figure 3.12 shows a message format in TraCI.

Mobility management is done by planning the movements of the nodes or vehicles in each time interval by the OMNeT++ network simulator. This fits perfectly with the operation of SUMO, because this simulator also handles simulation in discrete time. The state machine that includes the modules present in both SUMO and OMNeT++ is presented in Figure 3.13. These modules have the ability to store any command for execution in a certain time interval, and thus synchronize correctly. Thereby, OMNeT++ sends the assigned commands at each time instant to SUMO, and the traffic simulator starts the corresponding simulation iteration.

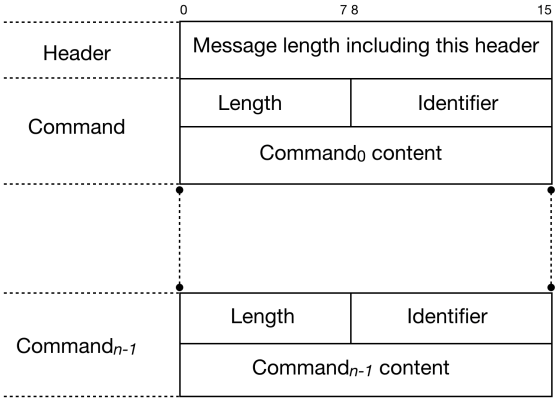


Figure 3.12: TraCI message format.

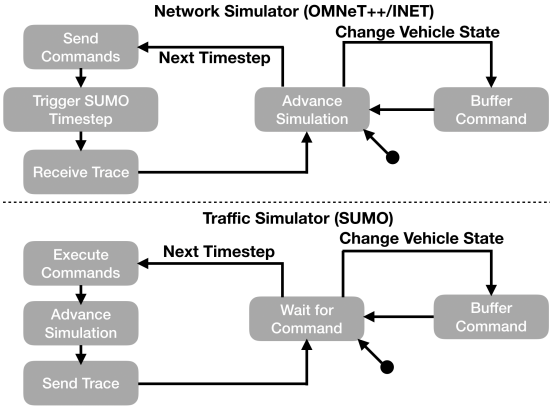


Figure 3.13: State machine of SUMO and OMNeT++.

At the end of the traffic simulation, SUMO returns a set of commands to OMNeT++, including the position information of the vehicles. Based on the mobility traces it receives, this network simulator introduces new vehicles, eliminates those vehicles that have completed their trips or have arrived to their destination, and continues to handle the remaining vehicles according to SUMO.

Additional vehicle travel statistics.

In our study, it is very important to have additional information about each vehicle that is in the simulation, such as travel time, the time a vehicle reaches its destination, and the segments crossed by the vehicle, with their respective speed

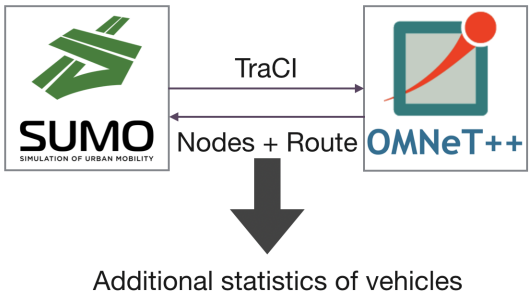


Figure 3.14: Joint vehicle management combining SUMO and OMNeT++.

limits. Therefore, with these additional statistics, we proceed to optimize vehicular traffic. To obtain these statistics regarding vehicles in a simulation of the city of Valencia, certain modules had to be modified within the TraCI architecture. In particular, we used the scheme shown in Figure 3.14, but we focused on the module *TraCIScenarioManagerLaunchd*, which allows us to create and move the vehicles controlled by TraCI.

This new implementation extracts various information from the vehicles simulated in a scenario. The first data obtained are the IDentification (ID) of those vehicles that have arrived to their destination, the following are the times of departure and arrival, along with the total travel time; other data include the total distance traveled by the vehicle with its corresponding average speed. However, to achieve traffic optimization it becomes necessary to extract more information, such as the segment of street traversed by a vehicle with its respective time of entry and exit, and finally the amount of vehicles ahead of the vehicle when it enters the segment studied. Nevertheless, this extracted information must be supplied to a file, so several process are inserted in TraCI source for performing the

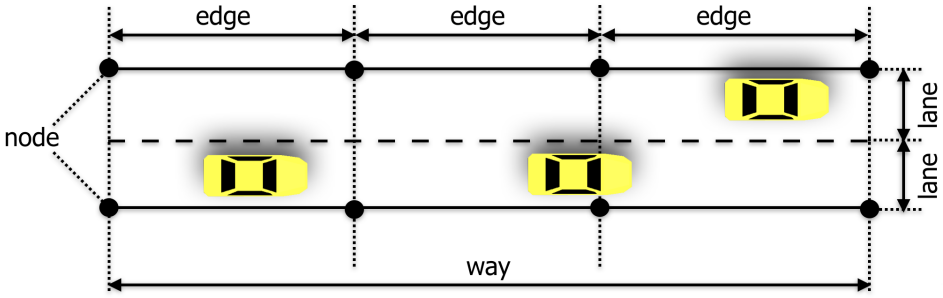


Figure 3.15: Structure of a street in SUMO and OMNeT++.

corresponding calculations. This is done every time a vehicle is inserted in the simulation and extract all traffic information such as departure and arrival time of the vehicles, vehicle speed in every street, moment of time entering and leaving a street, number of vehicles ahead of a vehicle, etc. Since the information corresponding to the time given by the simulation always starts from zero, we have to take into consideration the departure time of the vehicle, and determine the difference between the arrival time and the departure time when the destination is reached.

For a better understanding of the concepts related to the parts of a street or segment as adopted by the network simulator and the traffic simulator, we present an illustrative scheme in Figure 3.15.

3.3 ABATIS route server

3.3.1 Introduction

Currently, vehicular route servers rely on locally stored static information, which is used to calculate the requested routes. There are several commercial and free solutions including TomTom and Google Maps Navigator. The first deficiency of those systems is that they combine historical data with data sent by drivers' smartphones, portable browsers, collaborative social networks or any reporting application, meaning that users must constantly send reports about where they are, and how long it takes to get from one place to another. Such approaches are not effective if the users do not share this information to protect their privacy, or due to regulatory conditions such as the General Data Protection Regulation (GDPR). Besides, the second deficiency is that both Google Maps and TomTom perform only short-term predictions based on past and current status, but do not act as centralized controllers for all the traffic in a city. Thus, they cannot perform medium-term predictions nor balance traffic. So, we aim at a solution that is powered by the data coming from the induction loops deployed on the streets of Valencia, and made available by the City Hall, to avoid relying on any smart device as the primary data collection source, as in the case of Google Maps or TomTom.

Thus, a route planning server called ABATIS [17] was devised in order to provide route recommendations through a client-server interaction, and whose main objective is to offer the best possible routes, taking into account the cost of traveling on each street segment. ABATIS includes a characterization of the different streets of a city in terms of travel time depending on the vehicle load, thus allowing us to predict the traffic distribution.

http://jlzrou.grc.upv.es:5000/route/v1/driving/lon1,lat1;lon2,lat2.json?steps=true&overview=full&geometries=geojson

Figure 3.16: Example of a request to the ABATIS server.


```
{"waypoints":[
    {"location":[-0.372507,39.468762],"name":"Carrer de Colón"},
    {"location":[-0.350881,39.473339],"name":"Carrer del Doctor Vicente Pallarés"}],
"routes":[{
        "distance":2985.5,
        "duration":497.1,
        "weight":497.1,
        "geometry":{

            "coordinates":[[-0.372507,39.468762],[-0.372877,39.468492],[-0.373737,39.467889],[-0.372931,39.466708],,
            [-0.35567,39.468677],[-0.355362,39.469293],[-0.354771,39.470481],[-0.354512,39.471015],[-0.354376,39.471283],
            [-0.354167,39.471694],[-0.353673,39.472737],[-0.353068,39.473967],[-0.352456,39.473797],[-0.352193,39.473727],
            [-0.35198,39.473664],[-0.351213,39.473435],[-0.350881,39.473339]]}
    }],
    "code":"Ok"
}
```


Figure 3.17: ABATIS response in JavaScript Object Notation (JSON) format.

3.3.2 Server API

The server itself responds to route requests via an Hyper Text Transfer Protocol (HTTP) connection, which includes an Internet Protocol (IP) address, the listening port for the requests, the geographical coordinates for the starting and destination points, the type of service offered, and the output format.

For the case of the city of Valencia, any vehicle can send requests to our ABATIS route server by adding in the Uniform Resource Locator (URL) the geographic coordinates of both source and destination, as well as other additional parameters to offer more information to the client, as shown in Figure 3.16.

3.3.3 Output format

In Figure 3.17, we can observe an answer as returned by ABATIS in the JSON format. It contains information about the route that a vehicle will follow as a sequence of geographical coordinates, along with parameters such as:

- Waypoints: they are the geographical coordinates of the start and end points of the route with the corresponding street name.
- Routes: It is the summary of the calculated route, and includes the total distance and the trajectory time that has been calculated without any traffic (free-flow conditions).
- Route geometry: In general, it delivers the routes in an encoded format through the polyline coding algorithm; it encodes the geographical coordinates to provide a binary value as a series of American Standard Code for Information Interchange (ASCII) codes. The server offers the possibility to

deliver the routes without encoding them just by adding the "overview=full" parameter to the URL.

- Code: Indicate the status through a text string that can be "Ok" or "Cannot find route between endpoints".

Table 3.2: Output files of the ABATIS extract tool.

Metafile	Description
*.edg_nodes	Uncontracted edge-based graph used as input file for contractor the tool.
*.edge_penalties	Lookup file that contains static edge penalties.
*.turn_weight_penalties	Metafile that contains turn weight.
*.turn_duration_penalties	Metafile that contains turn duration.
*.edge_segment_lookup	Lookup file that translates between OSM IDs and internal segment IDs.
*.enw	Metafile that contains edge weights of the compressed EdgeBasedNodes.
*.geometry	Metafile used for API responses returned in route geometry format.
*.nbg_nodes	Contains the original graph nodes including coordinates and OSM node IDs.
*.edges	File with original graph edges in compressed geometry.
*.fileIndex	File index that contains leaves of the R-Tree that are loaded to memory on-demand.
*.icd	File with representations of all intersections for routing purposes.
*.names	Metafile with information about street names and index.
*.properties	Lua profile file properties that affect run time queries.
*.ramIndex	File with index of the R-Tree for segment lookups.
*.timestamp	Contains the timestamps of the OSM extract.
*.tld	Contains a map of turn lane tuples and IDs.
*.tls	Contains turn lane descriptions which are stored as a vector of masks.
*.osrm	Original filtered graph data such as nodes and edges for route calculations.
*.restrictions	Conditional restrictions such as the direction of the streets, which are then analyzed and applied on queries.
*.cnbg_to_ebg	Contains mappings from node-based graphs to edge-based graph.

3.3.4 ABATIS tools

To improve the performance of queries, and for a live-update of the traffic data on the route server, a modification of Dijkstra's algorithm called Multilevel Dijkstra (MLD) is used. Notice that our modified algorithm is aware of this multilevel partitioning by using precomputed overlay cells to heuristically speed up the computation [31]. For this reason, we present the tools that the ABATIS route server needs to complete our work.

Extract

Through this tool, the ABATIS server obtains information from the city of Valencia using as input a file with XML format that contains a large amount of data required to perform route calculations. In addition, the data does not conform to a strict standard, and important information can be described in several ways. So, it is necessary to extract the routing data in a standard format, as performed by this tool. In particular, it analyzes the contents of the exported file and writes routing metadata. Table 3.2 presents the files generated by this tool.

In addition, this tool uses Profiles that are used during this process to consider whether elements like private roads, pedestrian paths, barriers, etc., have to be taken into account when determining the best routes.

Table 3.3: Output files of the ABATIS partition tool.

Metafile	Description
*.cells	This file incorporates the cell data structure for the MLD.
*.partition	This file includes the partition information for the MLD.

Partition

This tool allows the graphs extracted from the streets to be divided into cells in order to perform routing calculations more efficiently and quickly. The files produced by this tool are presented in Table 3.3.

Customize

This tool gives the possibility to modify the weight of each of the cells that have been created by the partition tool. Thus, for a periodic traffic update, this tool is used together with an additional Comma-Separated Values (CSV) file that specifies the weights of the edges to be modified.

This traffic update file must be composed of three columns containing the following format: i) the first column indicates the identifiers of the initial node of

the edge; ii) the second column must includes the identifiers of the end node of the edge; finally, iii) the third column indicates the speed (in km/h) comprised between the initial node and the final node, as shown in Figure 3.18

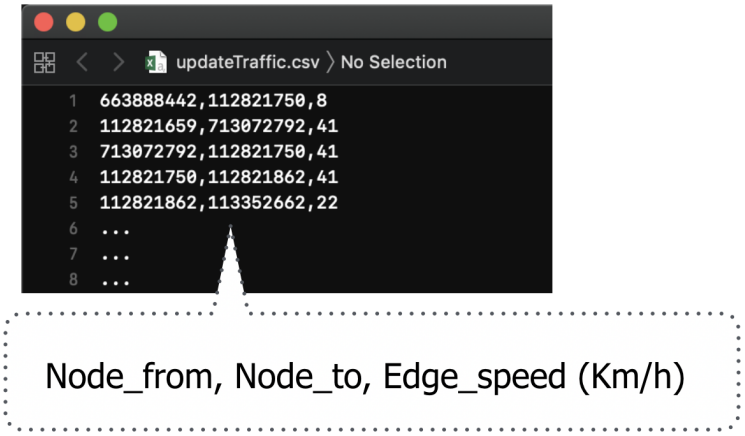


Figure 3.18: Traffic update file format.

In addition, output files that are obtained when executing this tool are detailed in Table 3.4

Table 3.4: Output files of the ABATIS customize tool.

Metafile	Description
*.mldgr	This file includes the search graph for MLD.
*.cell_metrics	This file incorporate all metrics of cells for MLD.

Datastore

This tool allows us to store all the information processed previously in the memory of the server in order to achieve greater efficiency and speed by requiring a query from a client.

In general, the server operating system allocates a block of memory when starting an application which is released when the application ends. So, this route server requires the use of shared memory that allows it to share data between several processes, remaining in the system until that block of memory is explicitly deleted.

The main objective of the use of shared memory is to update the traffic periodically, loading the different weights in the streets without the need to restart

the routes server. It is a substantial advantage for our work, because the traffic of Valencia will be constantly updated by each vehicle that requests information on the best route to its destination.

Routed

Finally, Routed is the tool used by the ABATIS server to attend incoming route requests. As parameters of this tool, the user has to consider the IP address with a listening port on the server, activate the shared memory functionality, and choose the algorithm to calculate optimal routes, which in our case is MLD.

Our approach requires periodic traffic updating. However, notice that the shared memory feature avoids having to restart the ABATIS route server every time. Thus, it simply executes the *Customize* tools to update cell weights and store these values again in shared memory using the *Datastore* tool. Figure 3.19 shows an initial execution scheme, and the traffic update scheme.

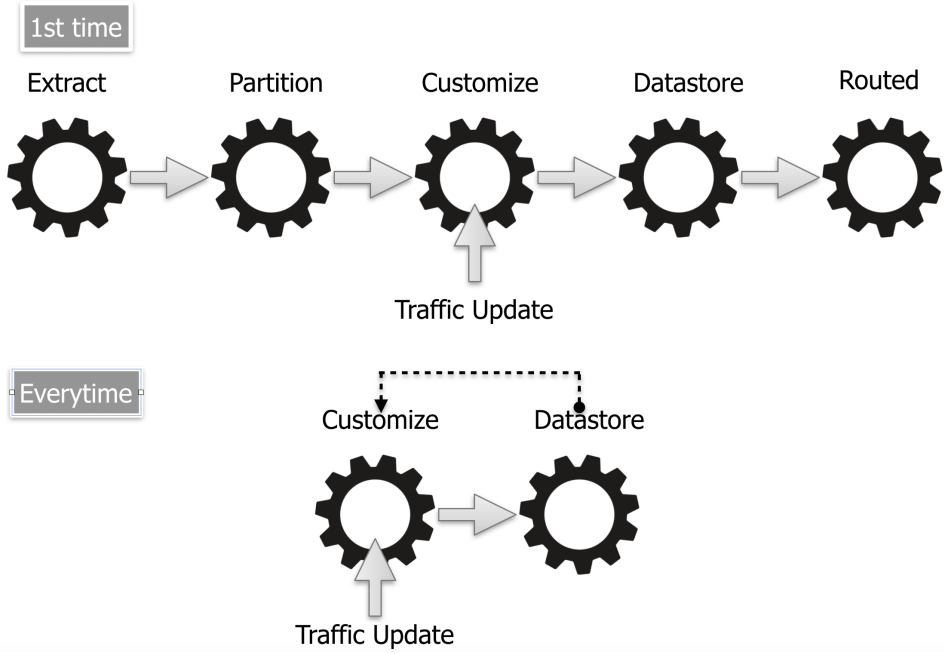


Figure 3.19: Process of execution and traffic updating in ABATIS.

Chapter 4

Traffic flow generation methodology

4.1 Introduction

In this chapter, we describe the procedure followed when attempting to obtain an accurate O-D matrix for the city of Valencia, Spain, as nowadays such essential data still remain unavailable. In fact, the only reliable data made available is generated by induction loops. Notice that many of the cities around the world rely on this method, and often lack any other sources of data beyond it.

The most critical data for simulation experiments to be representative is the number of vehicles that are injected into the network and their destination (O-D matrix). In this regard, the DFROUTER tool is able to generate data encompassing routes, and the number of vehicles associated with each route. Although not strictly an O-D matrix, such a matrix can be generated based on these data.

In this chapter we start by providing a general overview of the traffic levels in the city of Valencia, which is the scenario that is the basis of our study. This information was provided by the City Hall of Valencia. Afterward we describe the methodology followed to accomplish our goal of obtaining realistic traffic flow data, starting with an analysis of DFROUTER's output, and followed by the proposed heuristic, which will gradually reduce the number of vehicles passing through the induction loops, re-adjusting the number of vehicles to be injected in each segment until it is able to mimic the reference traffic, which is then validated against real data.

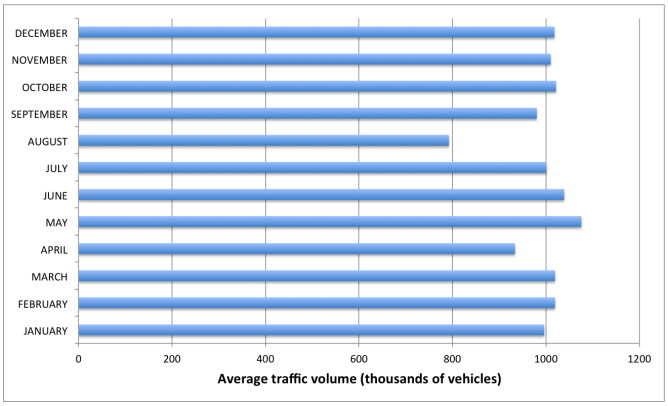


Figure 4.1: Average traffic volume in Valencia per month [17].

4.2 Overview of traffic conditions for the city of Valencia

Route planning solutions use fixed costs for different road segments, so selecting a particular origin and destination is an entirely deterministic process. In other words, there is a tendency to use the same streets or avenues repeatedly. However, the analysis of the traffic behavior in an urban environment shows that using the main streets or avenues during peak hours is usually not a good choice since the travel time is much higher compared to non-congested situations. Therefore, the calculation of the trip cost at different times of the day can benefit from a per-hour granularity analysis.

A study of the data provided by Valencia's City Hall shows that the main aspects that affect the travel cost for a certain street segment in an urban environment are:

- The traffic conditions are different for each street or avenue, requiring different cost models at different times of the day.
- Working days are characterized by traffic congestion patterns that differ from the behavior on weekends or holidays.
- Differences of season, being that people use more vehicles during the winter compared to summer.

In response to these factors, and focusing on a medium-sized European city such as Valencia, where traffic is measured daily in more than 500 streets and avenues, this would lead to more than 180 000 interpolation functions to describe the variation of traffic in each day.

To get an idea of the traffic flow, we will start by analyzing the monthly traffic, as shown in Figure 4.1. For our study we chose the month of November because

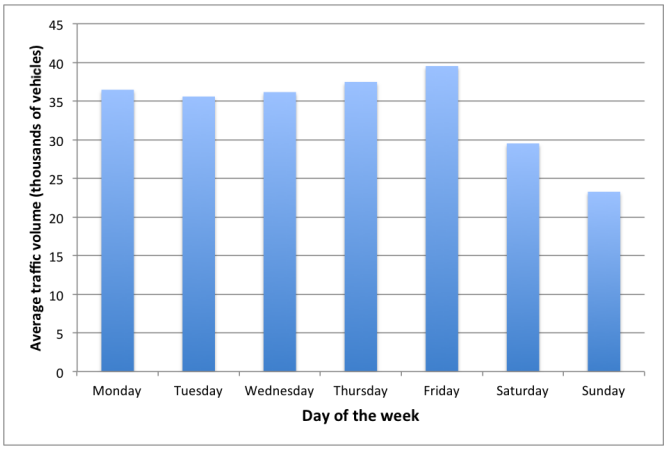


Figure 4.2: Average traffic volume in Valencia in a week [17].

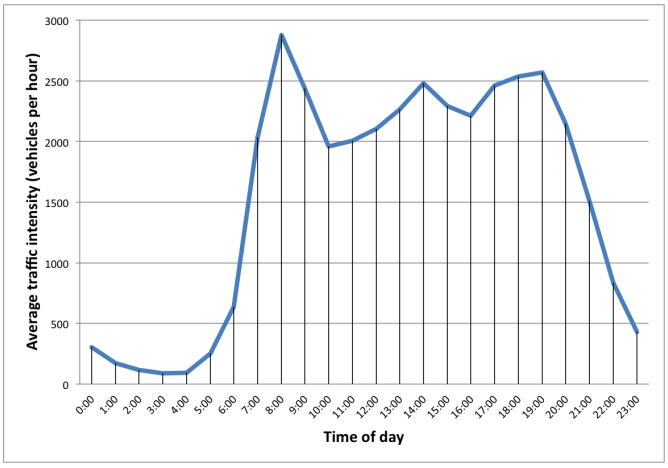
it does not have holiday periods, and it has values relatively similar to the other months.

Focusing on the variation of the traffic pattern during the week, we can see in Figure 4.2 that there are significant differences between working days and weekends. Thus, the pattern of traffic obtained for working days (Monday-Friday) encompasses 37,000 vehicles, and on weekends it drops to 26,000 vehicles.

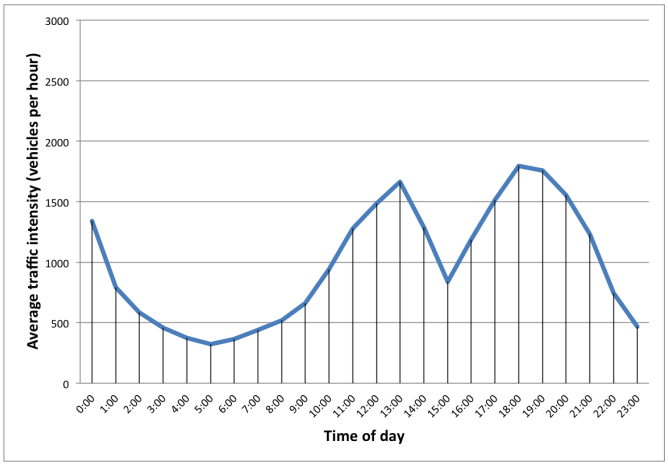
Due to these differences in terms of traffic volume, a typical Monday is chosen for our analysis, which, as shown in Figure 4.3, has the peak traffic volume between 8:00 a.m. and 9:00 a.m., when people usually go to work. Another peak hour is between 2:00 p.m. to 3:00 p.m., when people go out for lunch, and return to work, and the last peak hour is between 6:00 p.m. to 8:00 p.m., which means there is congestion when workers return home. We do not choose a weekend as Figure 4.3b, because the traffic intensity is not overly high as a typical Monday. From now on, we will choose the highest peak in Figure 4.3a.

4.3 DFROUTER output analysis

To evaluate how DFROUTER behaves using data from real induction loops, we use a database provided by Valencia's Traffic Department using the induction loop detector counts for the different streets and avenues of the city as input data. For our study, we chose the month of November because it has no holiday periods, and it has values relatively similar to other months. Based on the data obtained by Calafate et al. [17], we focused on a typical Monday, which is representative of traffic behavior at rush hour from 8:00 a.m. to 9:00 a.m. on business days, with 520 induction loop detectors deployed throughout the city.



(a) Monday.



(b) Sunday.

Figure 4.3: Average daily behavior for different days of the week [17].

To obtain the traffic characterization for our network in terms of vehicles injected and their respective routes, we followed the four-step procedure defined in Section 3.1.3 for the DFROUTER tool. In particular, the reference traffic flow data based on real induction loop measurements is used as a reference, being compared against the data regarding virtual induction loops following simulation measurements. After completing this process, we noticed some mismatches between the generated traffic and the original data, meaning that DFROUTER does not accu-

rately match the real traffic in the city of Valencia for the time period specified. In particular, the number of vehicles circulating in the city using DFROUTER is 138.9% greater than the overall number of vehicles detected by induction loops in the reference data. This happens because DFROUTER only knows the number of vehicles that pass through an induction loop detector, but it does not know the precise number of vehicles that have to be injected into the vehicular network to obtain that number. Furthermore, the traffic of different routes can converge to the same induction loop, and, therefore, it becomes difficult to predict in advance the actual number of vehicles passing through the different induction loops without actually running the simulation. In the next section, we propose a heuristic to correct this error and provide a more realistic traffic characterization.

4.4 Proposed Iterative Heuristic

Considering the mismatch between DFROUTER's output and the real data described in the previous section, we now propose an adjustment heuristic to compensate for this error, and make the number of vehicles in our simulation environment as realistic as possible.

To achieve our goal we propose a heuristic called "iterative" that regulates global traffic by affecting all streets in a proportional manner, thereby allowing to adjust the traffic volume according to the reference data. In particular, since there are dependencies between induction counts on different streets and the traffic injected, an iterative process is required to find the best fit. The proposed heuristic is shown in Algorithm 1.

As input data we have the information concerning vehicular traffic at a specific time, available as detector counts, and the associated street segment ID.

An initial run of DFROUTER is required for obtaining routes and the O-D matrix using the reference induction loop data as input. The outcome of this first run returns an excess of 138.9% in terms of traffic volume, a value that is then refined using our proposed heuristic. In particular, it will iteratively adjust the number of vehicles injected in every street as provided to DFROUTER so that, when the output converges, the total number of vehicles accounted in the output of DFROUTER is similar to the reference value. For this purpose, given a maximum number of iterations $n_{max} \in \mathbb{N}$ and a maximum admissible error $\varepsilon > 0$, we will build a succession of intervals $\{[\varphi^n_{min}, \varphi^n_{max}]\}^k_{n=0}$ of traffic flow adjustment factors, with $k \leq n_{max}$ and $\left[\varphi^{n+1}_{min}, \varphi^{n+1}_{max}\right] \subset [\varphi^n_{min}, \varphi^n_{max}] \forall n$, as well as a succession $\{\varphi_n\}^k_{n=0}$ with $\varphi_n \in [\varphi^n_{min}, \varphi^n_{max}] \forall n$, so that, for each iteration n, we will modify the number of vehicles injected on every street s as input to DFROUTER, and denoted by $\tau_{s,n}$, by relying on Equation 4.1:

$$\tau_{s,n} = \lfloor \frac{\sigma_s}{\omega_s} \cdot \varphi_n \rfloor \tag{4.1}$$

where σ_s is the smallest number of vehicles detected among all segments of the street s, ω_s is the number of segments that conform street s, and $\lfloor x \rfloor$ represents the largest integer number that is less than x. This formula ensures that the load corresponding to a street, according to our induction-loop data, is evenly distributed among the different segments on that street. Then, we apply the traffic flow adjustment factor φ_n to each street segment of that street.

Algorithm 1 Iterative heuristic.

Require: Road Network, flow, detectors files, n_{max} and ε
Ensure: Vehicle-Street-Segment-info file

$\alpha \leftarrow$ calculate reference number of vehicles
$\varphi^0_{min} \leftarrow 0$,$\varphi_0 \leftarrow \varphi^0_{max} \leftarrow 1$,$\tau_{s,0} \leftarrow \lfloor \frac{\sigma_s}{\omega_s} \cdot \varphi_0 \rfloor$
Process input files with DFROUTER
$n \leftarrow 1$
$\beta_1 \leftarrow$ Vehicle count per street ID
$\varphi_1 \leftarrow \frac{\alpha}{\beta_1}$
$\tau_{s,1} \leftarrow \lfloor \frac{\sigma_s}{\omega_s} \cdot \varphi_1 \rfloor$
Create a file with information about vehicles, segments and streets
Apply $\tau_{s,1}$ to all street IDs ($\tau_{s,n}$)
$\varphi^1_{min} \leftarrow \varphi^0_{min}$, $\varphi^1_{max} \leftarrow \varphi^0_{max}$
while $\left| \frac{\beta_n}{\alpha} - 1 \right| > \varepsilon$ **and** $n < n_{max}$ **do**
    Process input files with DFROUTER
    $n \leftarrow n + 1$
    $\beta_n \leftarrow$ Vehicle count per street ID
    if $\left| \frac{\beta_n}{\alpha} - 1 \right| > \varepsilon$ **then**
        if $\beta_n > \alpha$ **then**
            $\varphi^n_{max} \leftarrow \varphi_{n-1}$,$\varphi^n_{min} \leftarrow \varphi^{n-1}_{min}$
        else if $\beta_n < \alpha$ **then**
            $\varphi^n_{max} \leftarrow \varphi^{n-1}_{max}$,$\varphi^n_{min} \leftarrow \varphi_{n-1}$
        end if
        $\varphi_n \leftarrow \frac{\varphi^n_{max} + \varphi^n_{min}}{2}$
        $\tau_{s,n} \leftarrow \lfloor \frac{\sigma_s}{\omega_s} \cdot \varphi_n \rfloor$ to all street IDs ($\tau_{s,n}$)
    end if
end while

As the initial step in our iterative algorithm, we assign initial values to the different variables involved, as detailed in Equation 4.2:

$$\varphi_0 = 1,\ \tau_{s,0} = \frac{\sigma_s}{\omega_s} \cdot \varphi_0,\ \varphi^0_{min} = 0, \quad \varphi_0 = \varphi^0_{max} = 1 \tag{4.2}$$

Then, a relationship between the number of vehicles per street ID and the

reference value (φ) is calculated according to Equation 4.3:

$$\varphi_1 = \frac{\alpha}{\beta_1} \tag{4.3}$$

where α is the actual total number of vehicles passing through all street IDs (reference value), and β_1 is the simulated number representing the total amount of vehicles passing through all street IDs, as provided by the initial execution of DFROUTER. In our case, $\alpha = 409\,499$, $\beta_1 = 978\,339$, and, therefore, $\varphi_1 = 0.41857$ (see Table 4.1). Note that we will always have $\alpha \leq \beta_1$, although the value of β should converge to α.

For the iterations that follow ($n > 1$), the following equation applies:

$$\varphi_{n-1},\, \tau_{s,n-1} = \frac{\sigma_s}{\omega_s} \cdot \varphi_{n-1} \tag{4.4}$$

We then proceed to execute DFROUTER in a loop by successfully refining the input values, and obtain as output the average number of vehicles per street ID (β_n). The process continues converging until the error is close to 0 by obtaining a sequence of values for φ^n_{max} and φ^n_{min} based on the relationship between α and β, so that, if $\beta_n > \alpha$, we consider that $\varphi^n_{max} = \varphi_{n-1}, \varphi^n_{min} = \varphi^{n-1}_{min}$; in the opposite case, if $\beta_n < \alpha$, we instead consider that $\varphi^n_{max} = \varphi^{n-1}_{max}, \varphi^n_{min} = \varphi_{n-1}$. Then, a new φ_n value is obtained for the next iteration using Equation 4.5:

$$\varphi_n = \frac{\varphi^n_{min} + \varphi^n_{max}}{2} \tag{4.5}$$

A new value of $\tau_{s,n}$ (see Equation 4.1) can then be derived and used as input to DFROUTER. The new β value following each iteration n, denoted as β_n, will be considered final when the error ($\frac{\beta_n}{\alpha} - 1$) is below ε, or the maximum number of iterations is reached, as stated in Equation 4.6:

$$\left|\, \tfrac{\beta_n}{\alpha} - 1 \,\right| < \varepsilon \quad or \quad n = n_{max} \tag{4.6}$$

Experimental results in Table 4.1 show that, initially, the value of $\frac{\beta_n}{\alpha} - 1$ will fluctuate between positive and negative values, until convergence near to zero is achieved, as illustrated in Figure 4.4. For instance, from Table 4.1 we have that the DFROUTER output value β_2 is 439 738, which is greater than α, and $\frac{\beta_n}{\alpha} - 1 = 0.07384$. Since we want a small error, from interval $\left[\varphi^1_{min}, \varphi^1_{max}\right] = [0, 1]$ and φ_1=0.41857, the heuristic creates interval $\left[\varphi^2_{min}, \varphi^2_{max}\right] = [0, 0.41857]$ and value φ_2=0.20928.

Convergence is guaranteed in the scope of the proposed iterative heuristic, as we prove next:

On the one hand we have that

$$\forall_n \in \mathbb{N} \quad \left[\varphi^{n+1}_{min}, \varphi^{n+1}_{max}\right] \subset [\varphi^n_{min}, \varphi^n_{max}]$$

Moreover,

$$\forall_n \geq 2 \quad 0 <\mid \varphi_{max}^{n+1} - \varphi_{min}^{n+1} \mid = \frac{1}{2} \mid \varphi_{max}^{n} - \varphi_{min}^{n} \mid$$

$$= \frac{1}{2^{n-1}} \mid \varphi_{max}^{2} - \varphi_{min}^{2} \mid \leq \frac{1}{2^{n-1}}$$

Therefore,

$$0 \leq \lim_{n\to\infty} \mid \varphi_{max}^{n+1} - \varphi_{min}^{n+1} \mid \leq \lim_{n\to\infty} \frac{1}{2^{n-1}} = 0$$

Which implies that

$$\exists \lim_{n\to\infty} \varphi_{min}^{n} = \lim_{n\to\infty} \varphi_{max}^{n}$$

On the other hand, $\varphi_n \in [\varphi_{min}^n, \varphi_{max}^n] \ \forall_n \in \mathbb{N}$. Consequently, applying the sandwich criterion, we have that

$$\exists \lim_{n\to\infty} \varphi_n = \lim_{n\to\infty} \varphi_{min}^{n} = \lim_{n\to\infty} \varphi_{max}^{n}$$

Which in turn implies that

$$\forall_s \quad \exists \lim_{n\to\infty} \tau_{s,n} = \lfloor \lim_{n\to\infty} \frac{\sigma_s}{\omega_s} \cdot \varphi_n \rfloor = \lfloor \frac{\sigma_s}{\omega_s} \cdot \lim_{n\to\infty} \varphi_n \rfloor$$

Note that values $\tau_{s,n}$ must be integers because they represent numbers of vehicles. Therefore, the existence of the above limits implies that $\exists n_0 \in \mathbb{N}$ such that $\forall_n \geq n_0$ and $\forall s \ \ \tau_{s,n} = \tau_{s,n_0}$. That is, from iteration n_0 the input data to DFROUTER are always the same and, as DFROUTER is deterministic, its output data will also be always the same. This guarantees the existence of $\lim_{n\to\infty} \beta_n$, which does not necessarily match with α because both α and the limit are integer values, but according to the construction of the successions in the algorithm, it must be very close to α. Hence, given a reasonable upper bound to the number of iterations n_{max}, and a reasonable error $\varepsilon > 0$, it is expected that, for some $n \leq n_{max}, \left| \frac{\beta_n}{\alpha} - 1 \right| < \varepsilon$. Otherwise, the algorithm will stop at value $\beta_{n_{max}}$.

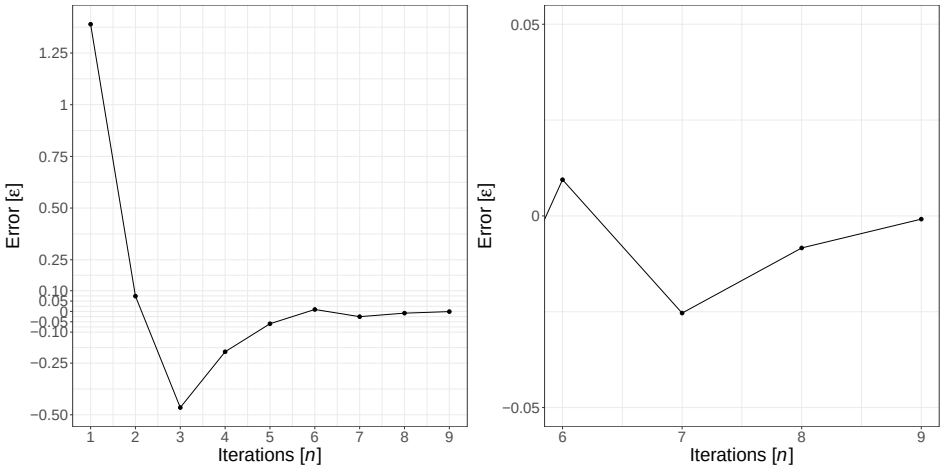
When the execution of Algorithm 1 ends, we generate two files that contain traffic information associated to the various street segment IDs. The first one is composed by a tuple of two elements, where each tuple includes:

1. Street segment ID.
2. Number of vehicles per segment ID.

The second file aggregates different street segment IDs belonging to the same street, and it includes a set of three elements:

Table 4.1: Iterative heuristic applied in the simulated traffic flow of Valencia city.

n	α	β_n	ε	φ_{min}	φ_{max}	φ_n
0	-	-	-	0	1	1
1	409 499	978 339	1.38926	0	1	0.41857
2	409 499	439 738	0.07384	0	0.41857	0.20928
3	409 499	219 071	-0.46503	0.20938	0.41857	0.31393
4	409 499	329 609	-0.19509	0.31393	0.41857	0.36625
5	409 499	385 108	-0.05956	0.36625	0.41857	0.39241
6	409 499	413 368	0.00945	0.36625	0.39241	0.37933
7	409 499	399 120	-0.02535	0.37933	0.39241	0.38587
8	409 499	406 074	-0.00836	0.38587	0.39241	0.38914
9	409 499	409 153	-0.00084	-	-	-



(a) Full chart.

(b) Zoom for near zero error values.

Figure 4.4: Estimation error evolution when increasing the number of iterations.

1. Street ID.

2. Associated segment ID.

3. Segment ID with the lowest number of vehicles, where the latter corresponds to the number of vehicles injected that start on that particular street.

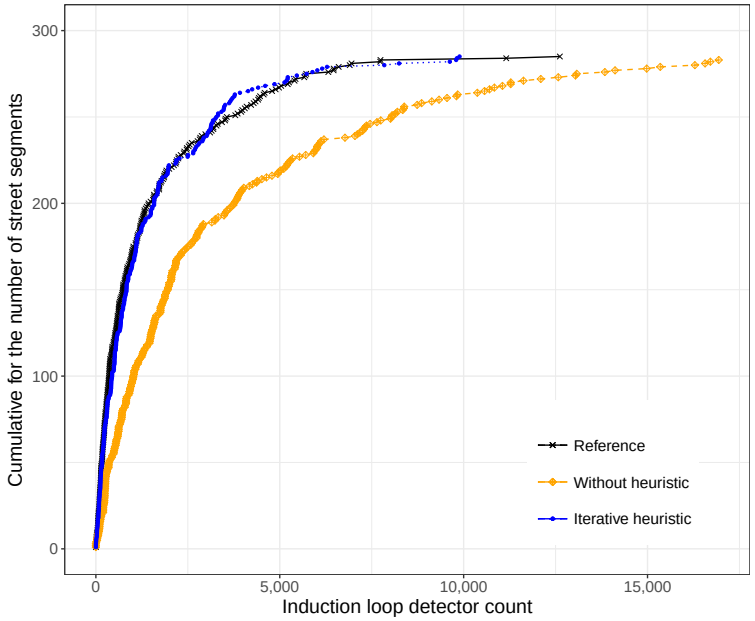


Figure 4.5: Adjustment of vehicles in Valencia using the proposed heuristic.

4.5 Goodness of fit achieved by the iterative heuristic

The methodology described above basically consisted in using our reference induction loop data as input to DFROUTER, and then refining output values in an iterative manner until the overall number of vehicles for the reference and generated data are quite similar. To validate the adequacy of this methodology, and check if the global optimization approach actually provided a similar distribution of values at finer levels of granularity, we obtained the cumulative distribution function of traffic for the different streets segments in the target city area. Figure 4.5 shows the obtained results, where we can observe that indeed the outcome of the iterative algorithm has a high resemblance with the reference data, achieving a behavior similar to the desired one in terms of the number of vehicles passing through the detectors for the different street segments of the city.

4.6 Test and simulation results

In this section, we compare our target city (Valencia) against other traffic mobility data for the cities of Cologne and Bologna existing through simulation to evaluate and compare the results obtained. For each city, we evaluate the traffic

source distribution, the traffic destination distribution, and the traffic dispersion distribution. Notice that, for the city of Valencia, we are using traffic flow definitions obtained according to the iterative heuristic defined in Subsection 4.4 with $\varphi = 0.38914$ and $\varepsilon < 0.0001$ in the 9th iteration, while the latter two are typical scenarios provided by the SUMO tool itself that will be used as reference. In particular, we want to determine whether the effectiveness of our proposed approach allows generating O-D and route information comparable to Cologne and Bologna, or if, on the contrary, results differ too much from these two reference cities.

Table 4.2 shows the main characteristics of the three different cities analyzed. In terms of target area, we find significant differences. In the case of Bologna, the target area is of only 2.34 squared kilometers. For Cologne we have the opposite situation, where the target area has a size of 595.9 squared kilometers. In the case of Valencia, the area analyzed covers the whole city (excluding suburban areas), having a size of 77.43 squared kilometers (intermediate case compared to others). In terms of street segment densities, they are clearly correlated with the overall area. If focusing on the average number of segments per km^2, though, we find that the density for Valencia and Bologna is quite similar, while Cologne has a much lower density since suburban areas are also included in the map. Concerning vehicle density, this is a metric that again has a clear relationship with the target area. In the case of Bologna, only a traffic-intensive area is analyzed. Cologne is in the opposite situation, covering a vast area, and for Valencia the whole city is included, although suburban areas are omitted.

Focusing on the generated traffic for the city of Valencia, the required information was obtained according to the procedure defined in this chapter. In the case of Cologne, data were generated by the DLR [32] using the SUMO mobility simulator. In particular, they relied on the DUAROUTER [6] tool to obtain routes through shortest path computation. Regarding the city of Bologna, its data were obtained from induction loops as well, as supplied by the municipality of Bologna. For performing route calculations, they get new routes assigned randomly according to a given distribution [33].

Our analysis focuses on three different metrics: traffic sources distribution,

Table 4.2: General statistic for the three target cities.

City	*Area* $[Km^2]$	*# street segments*	*# segment per* Km^2	*Vehicle density (per* km^2*)*
Valencia	77.43	11418	147.46	173.886991
Cologne	595.9	21953	36.84	2.31794261
Bologna	2.34	337	144.02	3 751.71

Table 4.3: Statistics about traffic sources in the cities under study.

City	*# Source positions per* km^2	*% of street segments used*
Valencia	3.36	2.28
Cologne	1.82	4.71
Bologna	5.13	3.56

traffic destinations distribution, and traffic dispersion distribution. In addition, for each of these metrics, we will analyze:

- The vehicle density per km^2.
- Percentage of street segments affected.
- The distribution along a map.
- The Cumulative Distribution Function (CDF) of vehicles per segment.

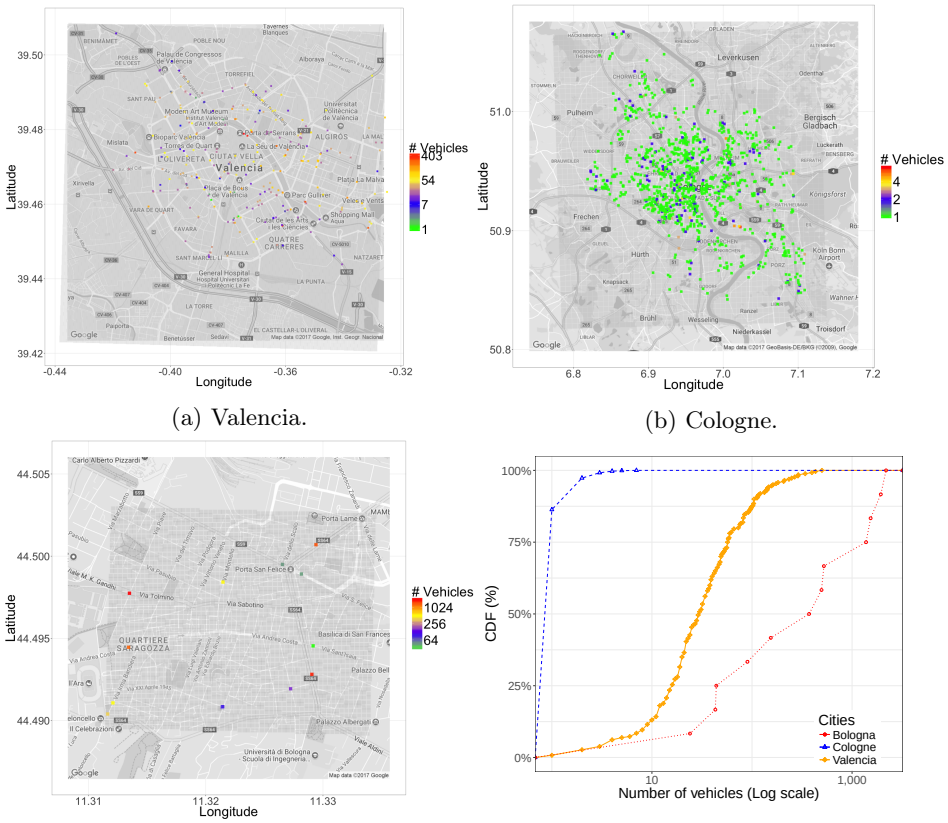
For the sake of a fair comparison, in the experiments that follow all scenarios were simulated using a simulation time of 900 s in their respective peak hours.

4.6.1 Traffic source analysis

We start our analysis by studying the location of the different traffic sources, their density per km^2, and how vehicles' starting points (i.e., sources) are distributed throughout the city map.

As shown in Table 4.3, Bologna has a vehicular density that is higher than other cities; however, notice that only a very small and busy area of the city is studied, as shown in Figure 4.6c. For Cologne we have the opposite situation compared to Bologna; Table 4.3 shows that it has a very low density of departure points, although we can observe a higher number of points in Figure 4.6b; this difference is expected as the target area is much greater, including more peripheral zones. Concerning Valencia, the number of traffic sources per square kilometer is a value in-between both reference cities. Figure 4.6a shows that these values are distributed throughout the city in a relatively homogeneous manner. Overall, we find that, in all cases, the number of source positions tends to be quite small.

Regarding the number of vehicles associated to each particular departure point, Figure 4.6d shows that, in Cologne, the number of vehicles associated to each source is in general very small (always less than 10); on the contrary, in Bologna, each traffic source may inject up to several thousand vehicles, which seems perhaps unexpectedly high. For Valencia, the spectrum of possible situations is much



(a) Valencia.

(b) Cologne.

(c) Bologna.

(d) CDF.

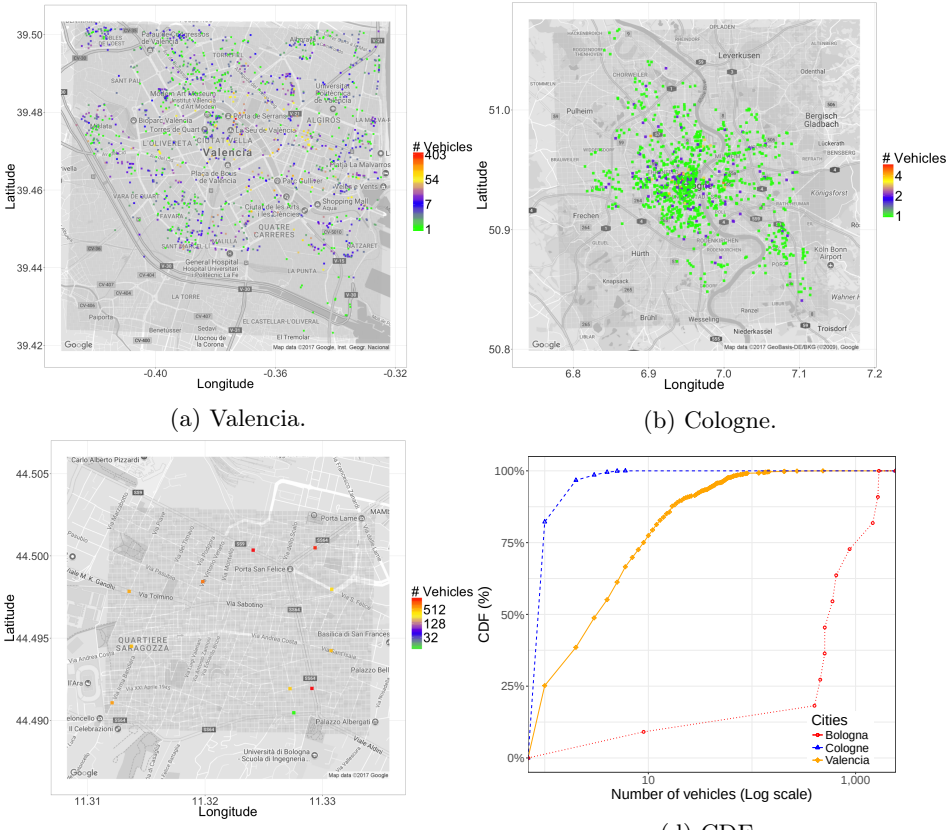
Figure 4.6: Geographical distribution of traffic sources (a, b, c) and CDF for number of vehicle per traffic source.

wider, where most traffic sources inject a moderate amount of vehicles (between 10 and 500 vehicles), including source points with only 2 or 3 vehicles. Overall, the obtained results for Valencia is more representative of a real metropolitan area, where highways may inject a high vehicle load into the system, while vehicles may also depart from other parts of the city as well

4.6.2 Traffic destination analysis

We now study the location of the different traffic destinations, their density per km^2, and how vehicles associated to the different ending points varies.

As shown in Table 4.4 and Figure 4.7, we find that Valencia is clearly the one



(a) Valencia.

(b) Cologne.

(c) Bologna.

(d) CDF.

Figure 4.7: Geographical distribution of traffic destinations (a, b, c), and CDF for the number of vehicles per destination position.

offering a richer set of destinations. In addition, although traffic destinations for Cologne and Bologna slightly decrease with respect to the number of sources, for Valencia this value substantially increases. This means that, for the specific case of Valencia, vehicle sources are often concentrated at some specific positions, like highway entrances, but then the destinations of those vehicles can vary substantially, as occurs in real life, which is indeed a good result.

Regarding the CDF for vehicle destinations, Figure 4.7d shows that Cologne maintains a distribution similar to the one observed for traffic sources (see Figure 4.6), while for Bologna a trend to concentrate destinations is detected (right shift).

Focusing on Valencia, we find that there is a clear left shift, which means that vehicles tend to complete their routes in a more heterogeneous manner, in accor-

Table 4.4: Statistics about traffic destinations in the cities under study.

City	*# Destinations per* km^2	*% of street segments used*
Valencia	17.13	11.61
Cologne	1.74	4.71
Bologna	4.70	3.26

Table 4.5: Statistics about traffic dispersion in the cities under study.

City	*# occupied street segments per* km^2	*% of street segments used*
Valencia	95.08	64.48
Cologne	12.88	34.97
Bologna	64.96	45.10

dance to the indexes and observations referred above, while maintaining locations that concentrate vehicle destinations. As expected, these locations are associated to the main highway exits.

4.6.3 Traffic dispersion analysis

Our first analysis concludes by studying traffic dispersion. In particular, we determine how the different street segments that conform the target maps are used by vehicles. For experiments to be meaningful, most of the main streets/avenues should have a non-zero traffic flow. So, we study the density of occupied street segments per km^2 throughout simulation time.

Focusing on the number of occupied street segments per km^2, both Table 4.5 and Figure 4.8 show that Valencia achieves a better occupation, which is associated to a better traffic distribution. For Cologne we have the opposite situation, being the number of occupied street segments per squared kilometer rather low. In addition, we find that the majority of street segments remains unused (65.03%). For Valencia this value is much lower (35.52%), in accordance with the percentage of secondary streets having negligible traffic. Regarding Bologna, its values are in the mid-range compared to Valencia and Cologne.

Concerning the CDF for the number of vehicles per street segment throughout the experiment, Figure 4.8d shows that, similarly to the CDF for traffic sources (see Figure 4.6d), Valencia is characterized by a wider range of situations, contrarily to Cologne (traffic is very sparse) and Bologna (traffic is very dense). In addition, if we compare the difference in shape between Figure 4.6d and Figure 4.8d, we observe that there is a general trend towards widening the range of values, and shifting towards the central values. So, in the case of Cologne and Bologna, we

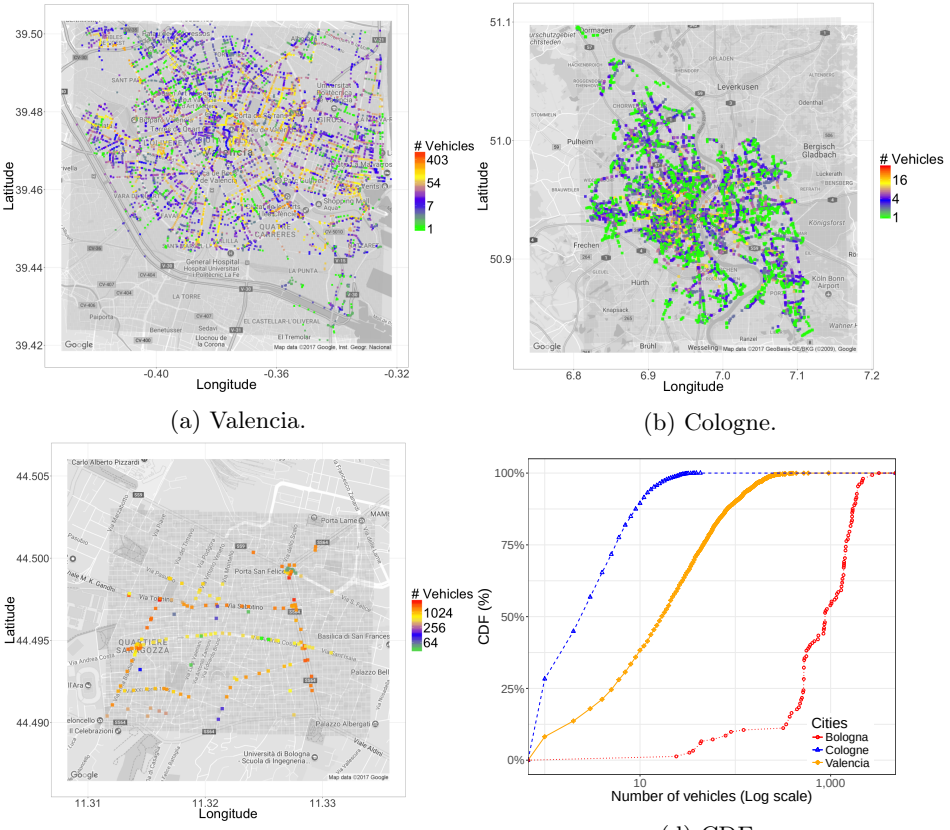


Figure 4.8: Geographical distribution of the street segments occupied by traffic (each points represents one full segment), and CDF for the number of vehicles per street segment throughout the experiment (d).

can now detect some traffic concentration effects (right shift of the curve), while in Valencia we have simultaneous traffic concentration and traffic dispersion effects, since the range of values now increases.

Overall, the analysis made shows that the procedure followed to generate O-D matrices for the city of Valencia, along with the mobility simulation to retrieve routes and the corresponding street occupation, clearly evidence that the results produced are exactly in-between both scenarios used as reference, as expected, being in accordance with the target area. In particular, they keep a clear relationship with the results for reference cities with wider areas (Cologne), and narrower areas (Bologna). Regarding the flows that are generated, we find that, for Valen-

cia, we achieve a much richer result, being that more possible flow destinations per squared km are contemplated, and that the percentage of segments used as destinations are also significantly greater than for the two other cities.

4.7 Summary

In this chapter, we proposed a procedure to obtain an accurate O-D matrix for the traffic of the city of Valencia that resembles the real traffic distribution, and that can be imported directly by the traffic simulator. Starting from real induction loops measurements made available by traffic authorities, we iteratively refined the output of the DFROUTER tool until the minimum possible error in the mismatch between the generated traffic and the original data was reached. When applying our iterative heuristic in the city of Valencia we obtained an error lower than 0.0001; in addition, we validated the results obtained through simulations against other traffic mobility data for the cities of Cologne and Bologna.

Overall, we observe a good dispersion of traffic through the different streets, which means that the traffic is flowing through a high number of street segments. Also, we find that there is a clear asymmetry between streets/segments with low and high traffic levels, as occurs in real situations.

Chapter 5

Traffic congestion analysis

5.1 Introduction

In the previous chapter, the traffic load for the city of Valencia has been properly adjusted and contrasted to other existing cities, allowing to achieve an O-D matrix that represents real traffic conditions at a specific day/time. However, traffic planning and optimization typically requires studying the impact of varying these conditions, especially if attempting to determine what degree of congestion is expectable if certain conditions cause additional traffic to circulate in the city. With this aim in mind, we will first detail how, based on that reference scenario, we can regulate the amount of vehicles in the city (λ) to generate different degrees of congestion, having the flexibility to congest the city according to any criteria. Specifically, our approach consists of attempting to inject an extra number of vehicles on each street segment (see Equation 5.1).

$$\tau_{s,n} = \frac{\sigma_s}{\omega_s} \cdot \varphi_n + \lambda_s \tag{5.1}$$

Once the simulation is executed under the new parameters, we then check the actual number of additional vehicles available in the system during the experiment, because not all vehicles can actually be injected if congestion is too high.

Since our control over additional traffic has a per-street granularity, below we detail two different methods for varying the traffic load, and study for each of them the impact of traffic congestion on different metrics of interest, including average vehicle speed, average travel time, and ratio of vehicle arrivals.

5.2 Uniform traffic load regulation

In this section we study the impact of uniformly varying the traffic load throughout the city of Valencia. To achieve this goal, our procedure was based on the reference traffic scenario for Valencia derived in Section 4, to assign a variable number of additional vehicles to be injected at each traffic source location.

Overall, we performed tests by varying the total number of additional vehicles injected into the system from 2 271 to 34 065.

Concerning the average vehicle speed metric, it was obtained by performing an arithmetic average of the speeds of the vehicles throughout the simulation experiment.

Figure 5.1 shows that, as expected, increasing the number of vehicles injected causes the average speed of vehicles to decrease due to the traffic jams that start to build up on the different routes. Also notice that, within the range of values tested, the correlation obtained is linearly inverse, meaning that congestion adjustments can be done in a straightforward manner.

We now proceed to study the average travel time of vehicles that have reached their destination; vehicles that failed to reach their destination during simulation time are excluded from our results. As shown in Figure 5.2, the average travel time increases in a linear trend as traffic in our scenario also increases. This trend is expectable, and very similar to real traffic conditions according to Highway Capacity Manual (HCM) [34].

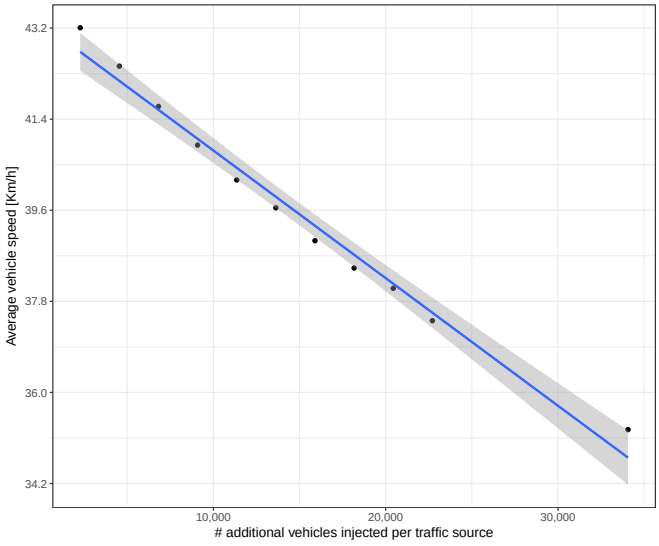


Figure 5.1: Average travel speed when uniformly varying the traffic volume.

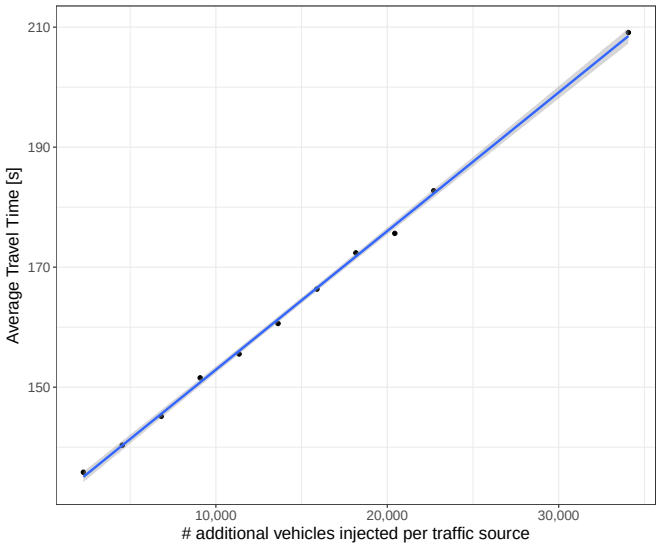


Figure 5.2: Average travel time when uniformly varying the traffic volume.

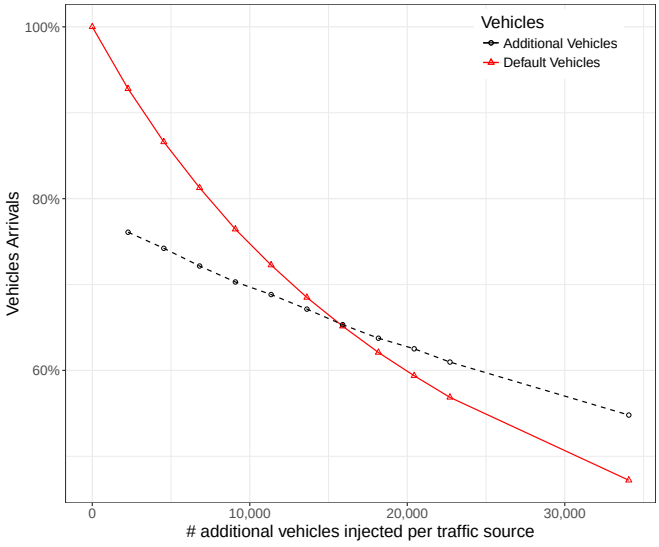


Figure 5.3: Vehicles arrivals (in percentage) when uniformly varying the traffic volume.

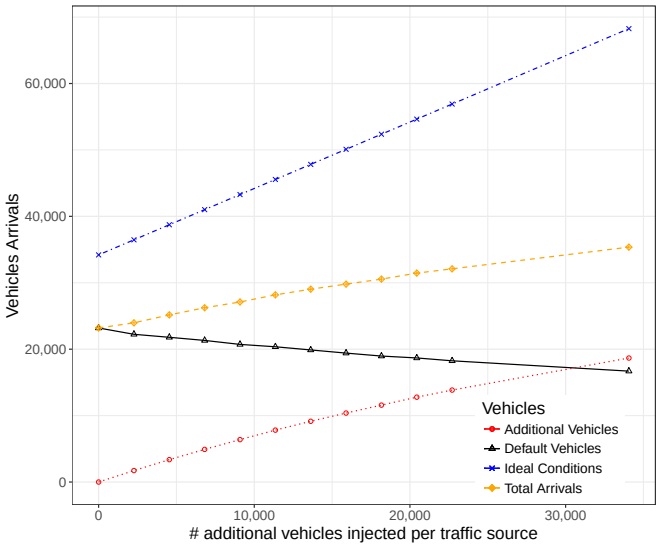


Figure 5.4: Vehicles arrivals.

Regarding vehicle arrivals, Figure 5.3 shows that, when injecting more vehicles in the road network of the city, all groups of vehicles (both default, and additional ones) experience a drop in their arrival ratio to their respective destinations, being this decay more pronounced for the *default* group of vehicles, as their number is actually larger.

Also, when focusing instead on the absolute value for the number of vehicles, Figure 5.4 shows that the number of vehicles in the *default* group experiences a constant drop, while additional injected vehicles experience an increase whose arrival rate slope slows down when reaching saturation conditions (number of additional vehicles injected similar to the default number of vehicles).

It is also worth observing how the total number of vehicles in the system varies, as well as its difference towards the ideal case, situation where the network capacity and conditions would allow all vehicles to reach their destination in a minimal time. Thus, the number of vehicles not arriving to their destinations is given by the difference between these two lines (ideal vs. total), a number that tends to increase more and more as system saturation conditions are reached.

To complete our analysis of the results obtained we now present the correlation between travel time and travel speed, followed by a travel time vs. travel distance analysis. Such correlation analysis aims at providing a greater insight into traffic behavior under congestion.

In terms of travel time vs. travel speed, Figure 5.5 shows that, when the average travel time increases, the vehicles' travel speed decreases, a situation expectable

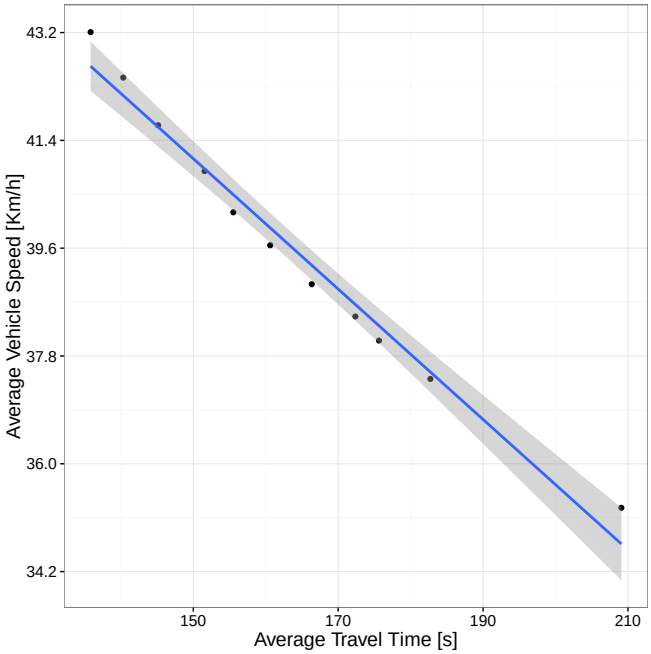


Figure 5.5: Correlation between average travel time and average speed.

for scenarios gradually becoming congested due to the presence of an excessive number of vehicles on the different streets.

Focusing now on the correlation between the actual distance travelled by vehicles according to their total time in the scenario, Figure 5.6 illustrates the behavior experienced under moderate traffic load conditions. As expected, distance and total time are clearly correlated, although we observe that the actual situation is quite complex, being that vehicles not arriving to their destinations were, in most cases, injected into the scenario in the initial simulation period, typically traveling a relatively small distance during that time. Concerning vehicles arriving to their destination, we find they were in some cases able to travel for long distances ($> 7\,km$ in some cases), although shorter trips are obviously prevalent. It is also worth pointing out that the maximum speed limits impose a minimum time to traverse a certain distance, which explains the slope of the points at the bottom of the chart.

Figure 5.7 depicts the situation under high traffic load conditions. We find that, on the one hand, point density has significantly increased (since each point represents a vehicle), and also that the correlation between time and distance now has a higher slope, meaning that more time is required, on average, to travel

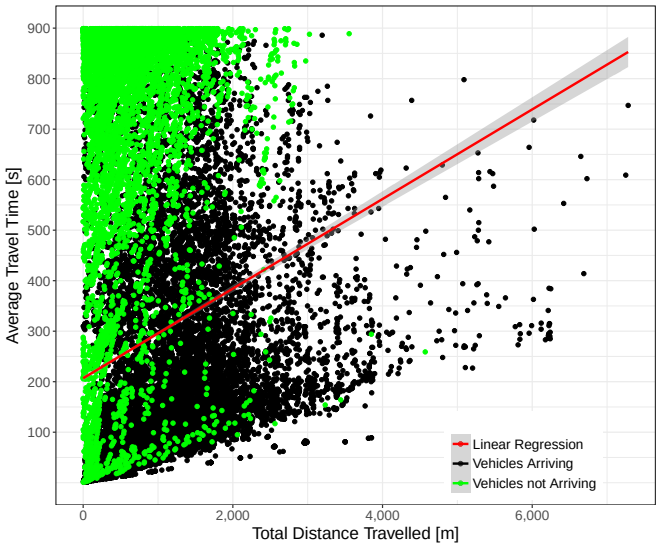


Figure 5.6: Time-Distance correlation under moderate traffic load conditions (total number of vehicles = 30 871).

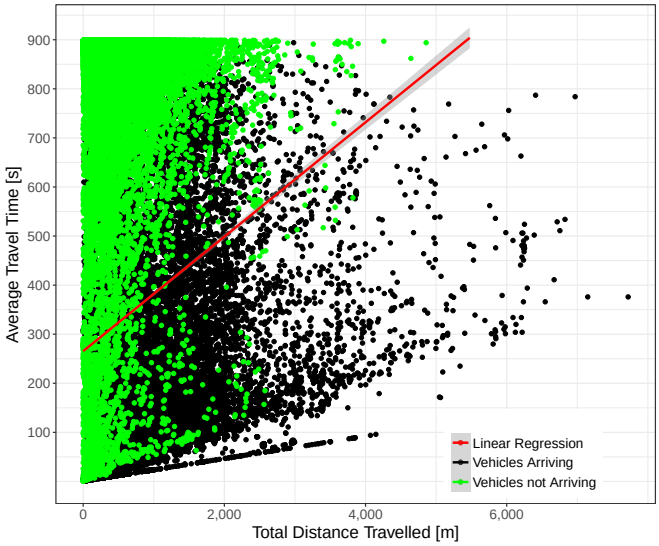


Figure 5.7: Time-Distance correlation under traffic saturation conditions (total number of vehicles = 53 463).

the same distance. Concerning the points distribution corresponding to vehicles arriving and not arriving to their final destination, we observe that both tend to concentrate nearer to the Y axis, a phenomena expectable due to congestion, although again we observe several cases of vehicles able to travel long distances.

5.3 Hotspot-based traffic load regulation

In this section our focus is now pointed towards situations where a public event is able to attract a high number of people to a very restricted area (e.g. football game), thus causing the city to experience a heterogeneous congestion effect. With this goal in mind, the Mestalla football stadium was chosen as the hotspot for our scenario. Notice that it is the biggest stadium in the city of Valencia, with a capacity of 54 000 spectators. Additionally, we find that up to 106 different routes pass near this stadium (within a 270 m radius). So, our strategy was to gradually inject vehicles into the system from about 100 to nearly 10 000; such vehicles depart from this area to congest traffic, as though it was a real situation taking place right after a sports event (see Figure 5.8). Notice that the additional vehicles will depart from the stadium to various end routes, thereby mixing with other existing traffic throughout the entire city.

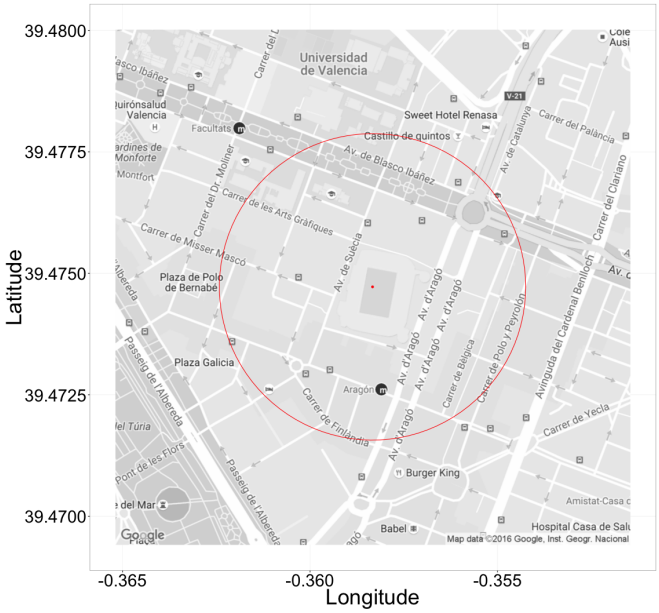


Figure 5.8: Area of our hostpot-based traffic.

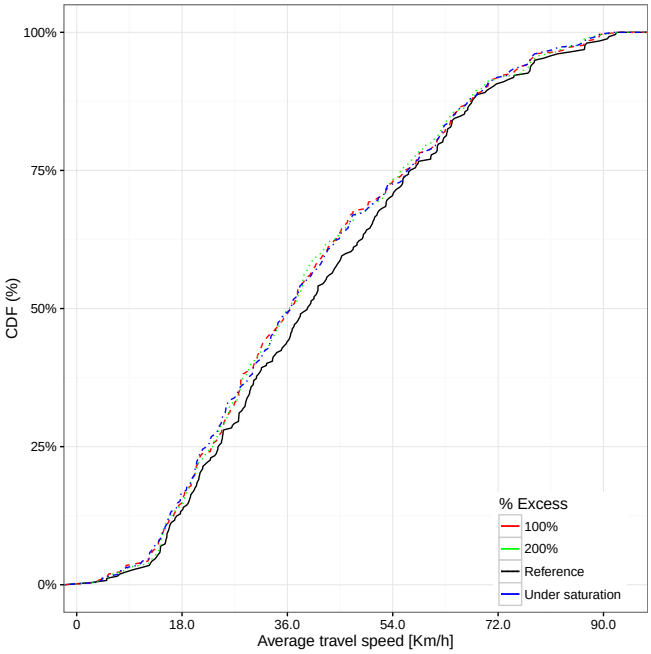


Figure 5.9: CDF for the average travel speed under different saturation levels.

Starting by observing the average travel speed distribution for all vehicles, Figure 5.9 shows that injecting more vehicles causes the average travel speed to be reduced; in particular, about 70% of the vehicles experience a travel speed below 54 Km/h.

Figure 5.10 shows instead the arithmetic average speed of all vehicles when increasing the number of additional vehicles injected into the system. We find that, as the number of vehicles inserted into the network increases, the average vehicle speed tends, in general, to decrease, although the degree of variability between experiments is high, as evidenced by the confidence envelope (represented in the figure as a gray shadow).

Focusing now on the average travel time associated to the different vehicles able to reach their destination, Figure 5.11 shows that an excess of 100% in the number of vehicles already has a quite significant impact on the average travel time distribution, being the curve in some cases very close to the saturation situation. Compared to the travel time CDF for all the vehicles (see Figure 5.12), we find that the overall differences compared to the reference case (non-congested) is basically the same for all saturation cases, meaning that the impact with 100% more vehicles departing from the hotspot area is already significant. If we compare Figure

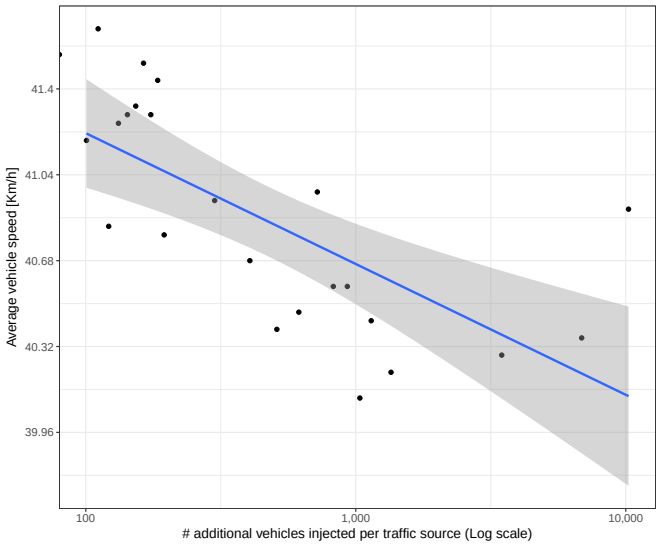


Figure 5.10: Average vehicle speed in different experiments.

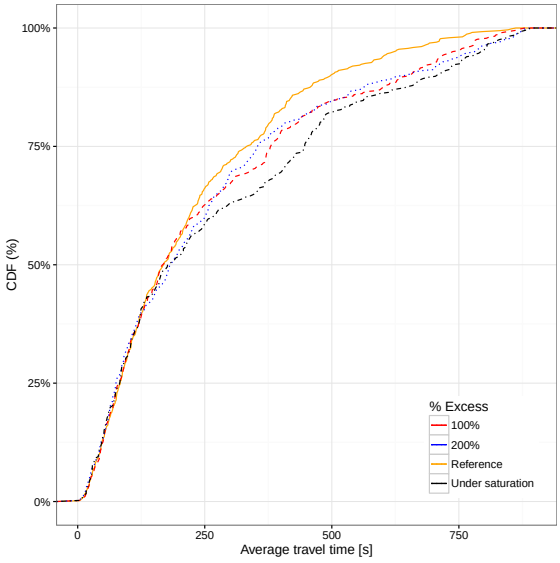


Figure 5.11: CDF for the average travel time under different saturation levels for vehicles arriving to their destination.

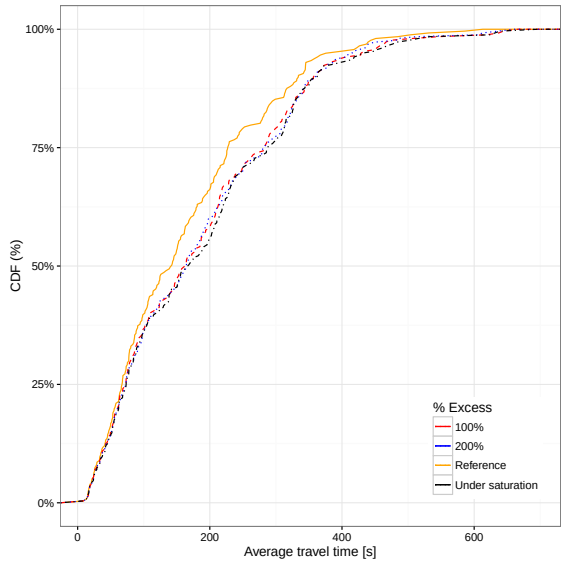


Figure 5.12: CDF for the average travel time under different saturation levels for all vehicles.

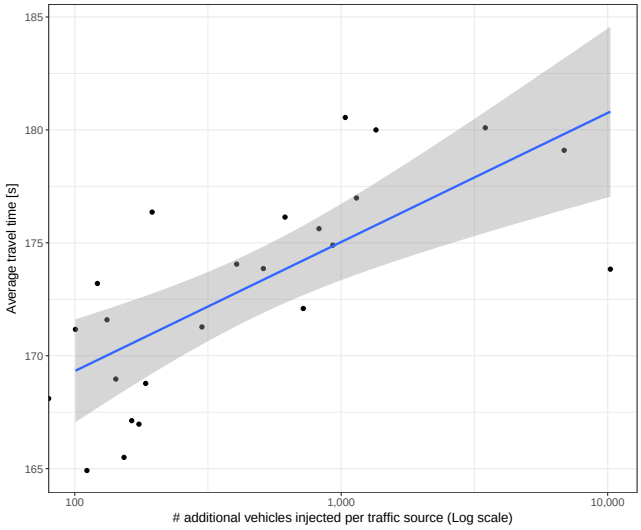


Figure 5.13: Average travel time in different experiments.

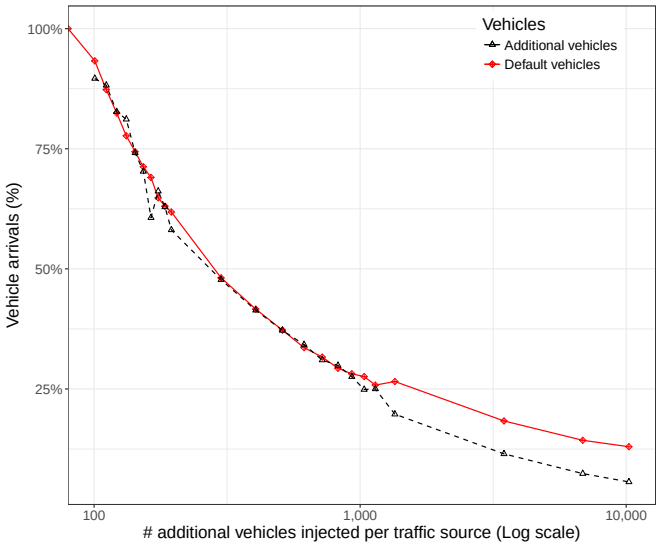


Figure 5.14: Vehicle arrival ratio when varying the number of additional vehicles injected per traffic source.

5.11 against Figure 5.12, we find that the differences for the former are more accentuated with respect to the latter.

This occurs since, when considering all vehicles, and not only those reaching their destination, we include many more vehicles, including those generated near to the end of the simulation time which, in general, are unable to reach their destination; this explains why differences become more subtle.

Figure 5.13 summarizes the correlation between additional injected vehicles and average travel time, evidencing that hotspot-based traffic injection is only able to provoke a moderate increase on the overall travel time of vehicles, as the impact in areas far away from the congestion point remains rather limited.

We now concentrate on the vehicle arrival ratio to see the percentage of vehicles that fail to complete their route within the simulation time. Figure 5.14 shows that, as the number of injected vehicles increases, the overall vehicle arrival ratio will experience a consistent decrease. When reaching more than 1 000 additional vehicles injected, though, the penalty experienced by additional vehicles is higher compared to the default vehicles in the scenario. This means that the congestion near the hotspot tends to become critical, but the impact on the rest of the traffic is not so dramatic.

Figure 5.15 provides further insight by representing the actual number of vehicles that arrive at their destination along with the total number of arrivals and the maximum theoretical case under ideal conditions. We can see that, as we increase

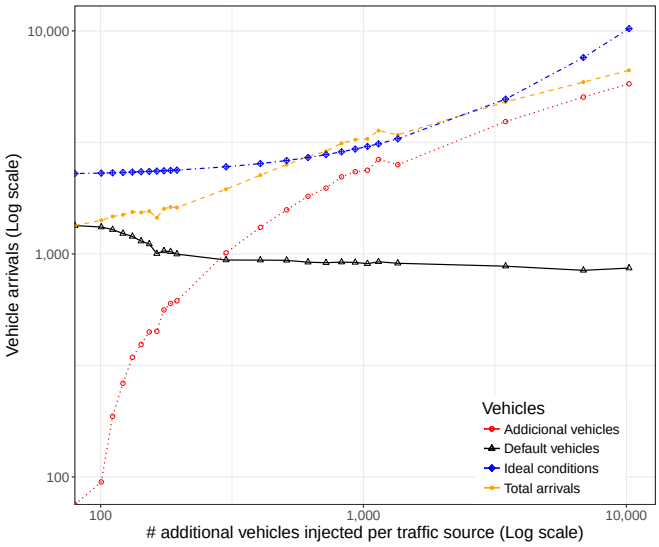


Figure 5.15: Vehicle arrival ratio when varying the number of additional injected vehicles.

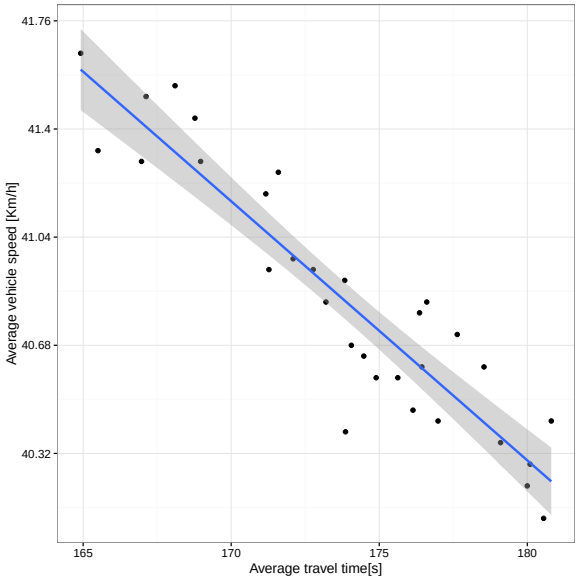


Figure 5.16: Correlation between average time and average speed.

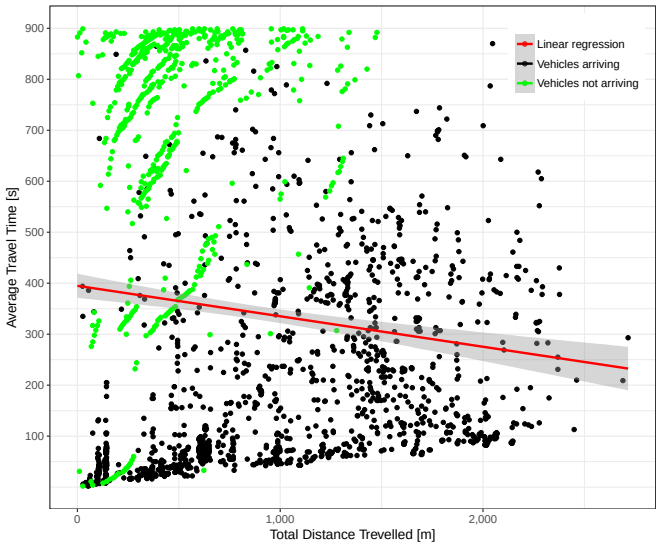


Figure 5.17: Time-Distance correlation under moderate traffic load conditions (total number of vehicles = 1 940).

the traffic load, the overall number of vehicle arrivals is able to increase only up to a certain extent (about 9 000 vehicles), beyond which saturation is reached. It is also interesting to observe that the additional injected vehicles will clearly represent the majority of the traffic when the number of additional vehicles injected per traffic source in greater than 20.

We now switch our focus to the correlation between average travel time and speed. Figure 5.16 shows, similarly to what we found before for the uniform scenario, that there is a clear inverse correlation with a thin confidence envelope, although now the speed variation range is much lower than before (less than 1 m/s), meaning that differences are mostly negligible.

We conclude our analysis by focusing on the relationship between total distance travelled and average travel time. Figure 5.17 shows the results achieved under low traffic saturation levels. In this case, we can see an unexpected result as the correlation line is negative. This occurs because many vehicles not arriving to their destinations, concentrated near the hotspot area, only travel for a short distance, thereby resulting in this bias. Overall, we can see that the distribution of vehicles arriving to their destination ranges from low to high distances, and from short to long travel times, while vehicles not arriving are concentrated near the upper edge, meaning that they were unable to reach their destination despite being present in the system from the beginning of the experiment in most cases.

Figure 5.18 shows the same correlation, now under high traffic load conditions.

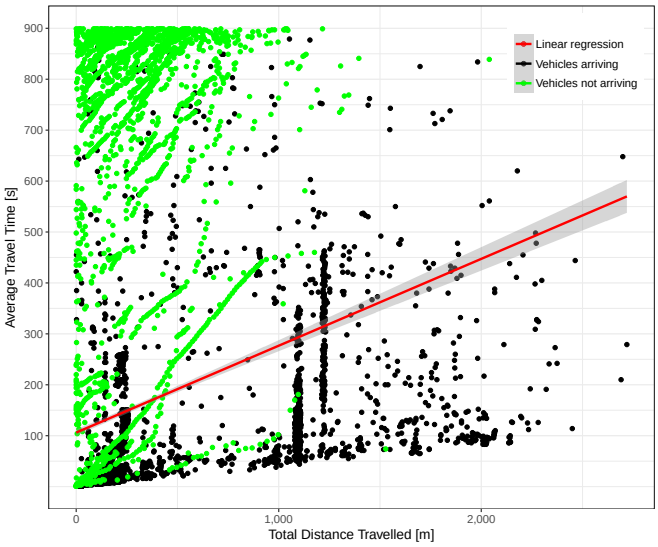


Figure 5.18: Time-Distance correlation under traffic saturation conditions (total number of vehicles = 9 985).

Compared to the previous figure, we can notice two main issues: (i) the correlation between total distance and total time is again proportional, and (ii) the vehicular density is higher, and mostly concentrated near the Y axis, similarly to the results obtained for the uniform traffic load scenario.

Overall, the results obtained confirm that the proposed approach allows: (i) to seamlessly regulate the amount of traffic in the network, in this case departing from a specific hotspot; (ii) to analyze the impact imposed on the overall traffic system; and (iii) to propose alternatives to make traffic flow more efficient, even in such extreme congestion situations.

5.4 Summary

In this chapter we proposed a strategy to allow us regulate the congestion levels in the city. Based on that strategy, we then analyzed the traffic behavior under different congestion conditions. Our analysis started by focusing on the whole city of Valencia, and then moving to a specific hotspot, where we could achieve even higher congestion conditions. Our experiments showed that injecting additional vehicles into the road network gradually increases the travel time and decreases the average speed of the vehicles, having a behavior that is mostly linear. In the next chapter we will focus on how to model such behavior.

Chapter 6

Traffic characterization and modeling procedure

6.1 Introduction

In this chapter, we describe the procedure followed to characterize the traffic in the city of Valencia, Spain, from a microscopic perspective, and starting from OpenStreetMap road layouts. In particular, our goal is to characterize individual street segments in terms of average travel times experienced by vehicles for different degrees of congestion, being the latter estimated based on the number of vehicles found ahead by a vehicle at the time it enters a segment. To achieve this goal, we found necessary to first perform some preprocessing, as in many cases the presence of micro-segments (streets unnecessarily partitioned into many short segments) impeded an adequate analysis of the behavior of vehicles traversing particular streets, as we can see in Figure 6.1. Then, we used the SUMO tool coupled with the OMNeT++ simulator to study the traffic flow for the entire city of Valencia based on a realistic traffic trace, as described in the previous section.

Throughout the chapter we will describe the methodology followed to characterize and predict traffic for the different street segments. Our proposed methodology is the following: first, we unify segments whenever required; next, we predict the number of vehicles in each segment; finally, we characterize the different street segments through a regression analysis for each street segments to properly obtain a relation between the number of vehicles ahead and the travel time for that segment.

Below, we detail the algorithms we proposed to unify segments, to predict the

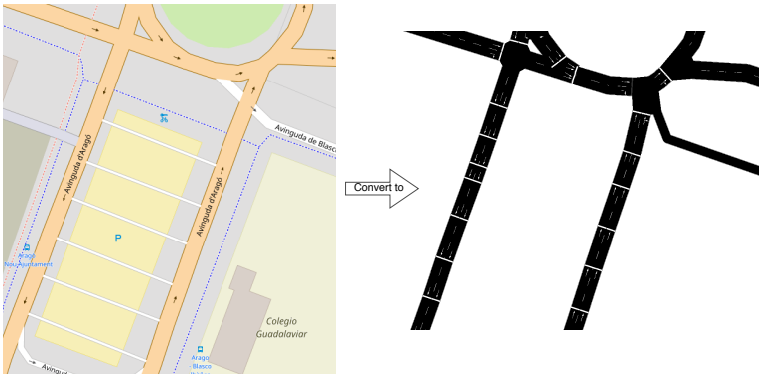


Figure 6.1: Example of unnecessary street partitioning.

traffic time, and to characterize the different segments according to vehicle travel times. Finally, by performing a clustering analysis, we were able to clearly identify three independent categories, whose characteristics are then properly discussed.

6.2 Unifying segments

Usually, when the city map is converted to a format accepted by SUMO for simulation, certain characteristics of the map must be eliminated, such as bicycle paths, pedestrian paths, train tracks, etc. This conversion has a drawback because it causes the streets to be intercepted by other ways, different from those used by vehicles, and the SUMO simulator acts by partitioning those streets. In many cases, the unnecessary partitioning of streets causes inconveniences such as:

1. Streets are partitioned into tiny segment sizes, often measuring less than 7.5 m (size of a vehicle plus inter-vehicular security gap).

2. Such small sizes do not allow us to characterize the segment profile correctly.

3. Inconsistent graphs are obtained when applying the regression analysis to predict traffic behavior.

To understand the solution adopted to address this issue, we should mention that the ID that represents a street segment is composed of two parts, where the first part is the code identifying the street, and the second part is the sequential code assigned by SUMO to each street partition. The proposed procedure tries to unify those street segments whenever possible if certain conditions are met. To achieve this, it is necessary to have a dictionary that stores all the segments without intersections, as well as a dictionary that stores all the connections of those

segments. Then, we must compare each connection of the different segments to determine whether they meet the conditions required to perform segment unification. In particular, the conditions that a set of segments must meet to be reunified are the following:

1. The street to be reunified must be a set of partitioned segments.
2. The adjacent segment should not have another segment that intersects it.
3. The street ID codes must be the same for segments to be reunified.
4. Segments to be reunified must have consecutive numbers in their sequential part of the ID.

In Figure 6.2 presents an example of the segments that meet the conditions mentioned above for the unification of segments. Once all these conditions have been met, at least two segments can be renamed and unified to achieve the correct prediction of the traffic, according to Algorithm 2.

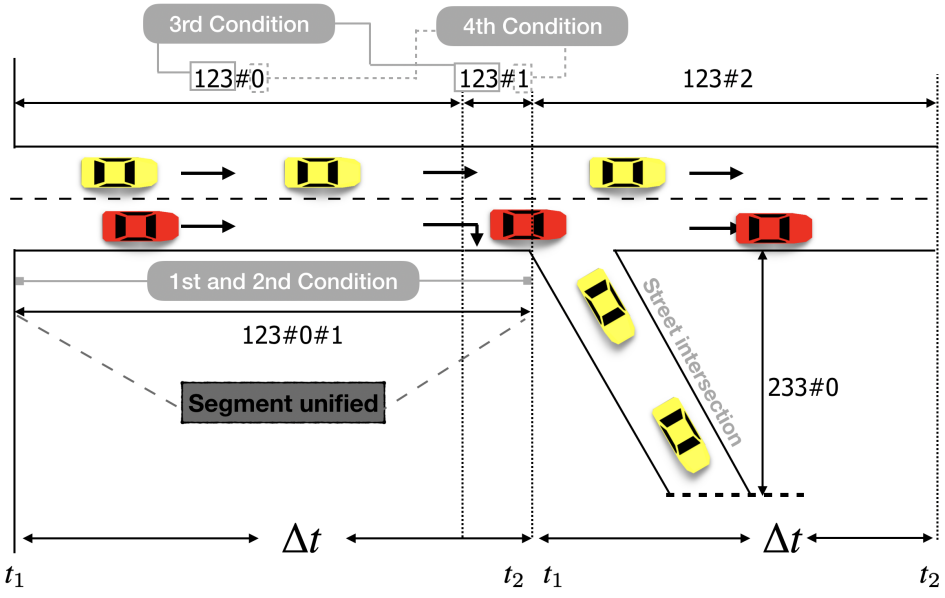


Figure 6.2: Example of the conditions for unify segments.

Algorithm 2 Segment reunification strategy.

Require: Road Network file, edges files
Ensure: Reunified segment file

```
edgeConnectedNoIntersection[] ← dictionary that stores all edges without intersections
for all edge in Road Network file do
    edge_id ← store the edge ID of the road network file
    connections[] ← dictionary that stores all connections for that edge id
    for all connection in Road Network file do
        connection_from ← store the edge ID(from) of the road network file
        connection_to ← store the edge ID(to) of the road network file
        if (connection_from = edge_id) and (connection_to not in connections) then
            connections[edge_id] ← connection_to
        end if
    end for
    if edge_id partition = TRUE then
        lenEdgeConnect ← length of dictionary in a specific edge ID
        street_id ← code of the street
        for i = 0 to length(connections[edge_id]) do
            if (lenEdgeConnect = 1) and (connections[edge_id][i] partition = TRUE) and (street_id in edge_id) and (street_id in connections[edge_id][i]) then
                edgeConnectedNoIntersection[edge_id] ← connections[edge_id][i]
            else
                edge_id ← has some intersecting segments, cannot reunify
            end if
        end for
    else
        edge_id ← is not split, reunification is unnecessary
    end if
end for
```

6.3 Per-Segment travel time prediction

In this section, our goal is to predict the travel time associated to each segment for different degrees of congestion, being the latter measured as the number of vehicles located in the segment just before a new vehicle enters it (ν_n).

We propose Algorithm 3 to achieve this prediction; in particular, we have to consider the input time (t^{ν}_{in}) and the output time (t^{ν}_{out}) of the vehicle in the

segment, as well as the number of lanes of the segment (l_n) where the vehicle is traveling.

Input times (t_{in}) for vehicles entering a segment on lane l_n are registered in matrix (l_n,t_{in}), while output times (t_{out}) for vehicles leaving the segment at lane l_n are then registered in matrix (l_n,t_{out}). To avoid erroneous values for the travel time prediction associated to each segment, it is necessary to perform a sorting of the matrix by t_{in}.

The number of vehicles in the segment just before a vehicle joins it (ν_n) will increase as long as t^{ν}_{in} is less than t^{ν}_{out}, and both refer to the same lane. Then, the travel time of each vehicle in the segment will be obtained (Δt).

As a final step, according to Algorithm 3, an average of the travel times ($\bar{\Delta} t$) associated to different degrees of congestion (number of vehicles in the segment before a new vehicle enters that segment) will be included in a file, along with the number of vehicles in that segment (ν).

Algorithm 3 Extraction of travel times vs. load samples.

Require: Reunified segment file, Segment-info files
Ensure: Statistical learning by segment files

1: **for** segment **in** Reunified segment file **do**
2: $segmentConnected[] \leftarrow$ vector that stores all edge ids connected
3: **for** s=0 **to** length($segmentConnected$) **do**
4: $segment\_info \leftarrow$ Read lines $segmentConnected[s]$ in Segment-info files
5: $segmentSorted[][] \leftarrow$ sort_by_t_{in}($segment\_info$)
6: **for** t_{in}=0 **to** length($segmentSorted$) **do**
7: $\nu_n \leftarrow$ number vehicles per segment in each lane
8: **for** $t_{out} = t_{in}$ **to** $t_{out} >= 0$ **step** -1 **do**
9: **if** ($segmentSorted[t_{in}][l_n] = segmentSorted[t_{out}][l_n]$) **and** ($segmentSorted[t_{in}][t^{\nu}_{in}] <= segmentSorted[t_{out}][t^{\nu}_{out}]$) **then**
10: $\nu_n = \nu_n + 1$ increase the number of vehicles if the condition is met
11: **end if**
12: **end for**
13: **end for**
14: $\bar{\Delta} t \leftarrow$ average of the travel times for different degrees of congestion
15: $\nu \leftarrow$ number of vehicles in the segment before a new vehicle enters the segment
16: **end for**
17: **end for**

6.4 Segment behavior characterization with polynomial regression

Once the process described above to estimate travel times in a segment for different degrees of congestion is completed, the next step is to characterize and classify the behavior of the different segments. In particular, we seek to determine the relationship between the number of vehicles in a segment (x values), and the average travel time of vehicles (y values) for each particular segment. To achieve this goal we perform regression to obtain the best curve fit describing the nonlinear relationship between segment congestion and travel time. In general, traffic theory considers that the relationship between traffic load and travel time tends to vary quadratically [35]. Thus, to perform the fitting, we used function $f(x) = ax^2 + t_{ff}$ in a first attempt. This way, the chosen expression is able to adequately represent this parabolic behavior starting from the free-flow travel time (no congestion), represented by constant t_{ff} in this function, and then increasing as the number of vehicles ahead in a segment (represented by x) increases.

Once the regression results for all the segments tested where obtained, we observed that the expected quadratic behavior was indeed taking place in many of the segments, although other special cases were also detected. Figure 6.3 presents different representative cases corresponding to the patterns we observed. The first class of polynomial regression curves, which illustrates the expected pattern according to traffic engineering theory, is shown in Figure 6.3a. As can be seen, when a vehicle entering a segment finds many vehicles ahead, it will, on average, experience much higher travel times, with differences up to 1000% being possible and expected. However, other patterns were also obtained, as traffic flow properties also cause other types of behavior to take place, especially when modeling a very large city such as Valencia. For instance, the second class of curve represents a behavior that is just the opposite compared to the previous one. As shown in Figure 6.3b, this kind of curve shows an increase followed by a decrease in the time traveled as we increase the number of vehicles ahead. Such behavior is explained by different factors, including the departure of vehicles from the segment, as they turn to join other segments, and, more important, the presence of traffic lights that tend to accumulate vehicles on the segment, being that vehicles finding many vehicles ahead usually means that the accumulation period was long, and the semaphore is about to turn green.

In addition to the two types of behavior described above, there are also other cases taking place, as exemplified in Figure 6.3c,d. Regarding the behavior observed in Figure 6.3c, we can see that the travel time remains constant regardless of the number of vehicles in the segment, typically meaning that there are no semaphores (areas in the periphery of the city), no junctions, and that the capacity of the segment is much higher than the number of vehicles detected during the simulation (typically multi-lane segments), and so congestion effects are not

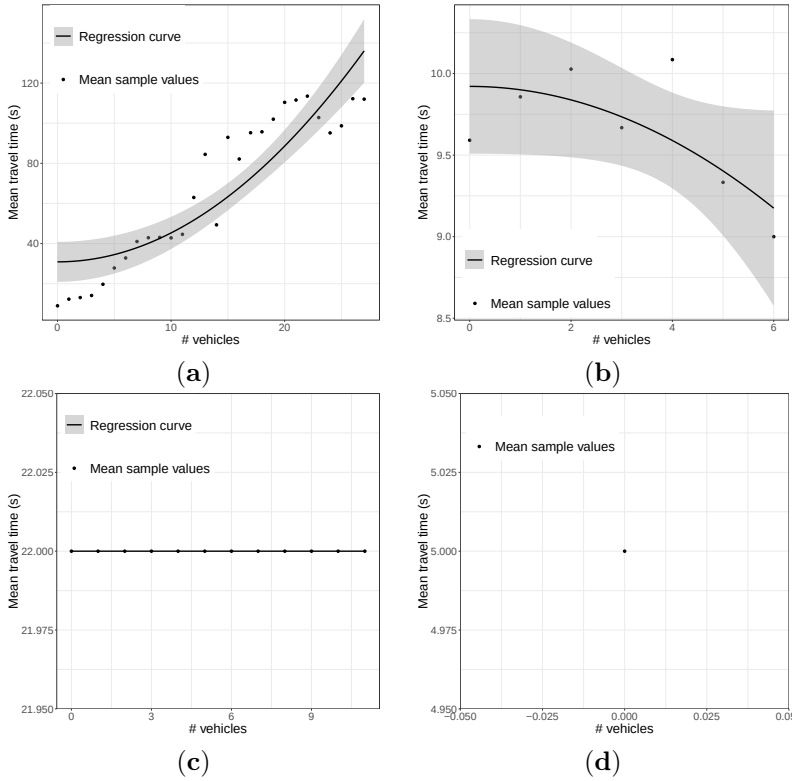


Figure 6.3: Segment Classification: (**a**) increasing; (**b**) decreasing; (**c**) constant; and (**d**) unique.

perceived. Finally, the behavior of the last group, as described in Figure 6.3d, corresponds to one-lane segments rarely visited by vehicles according to the traffic patterns used as input. Thus, we do not have enough information to characterize the behavior of the segment at higher loads, as the traffic flow levels are minimal.

Overall, we find that the characterization of the behavior of the segments in this scenario has not been done accurately in many cases where the vehicular load remained low. Thus, we deem it appropriate to saturate with additional vehicles the different segments of the city, and then improve our awareness regarding the behavior of all segments with the different degrees of congestion. In addition, we also want to know what happens with the second class of curve shown in Figure 6.3b, which represents an abnormal situation according to traffic theory. To achieve this goal, it is necessary to use Equation 5.1 presented in Chapter 5, to gradually increase the number of vehicles in our reference traffic scenario for Valencia until a higher saturation level is reached. This is achieved by assigning a

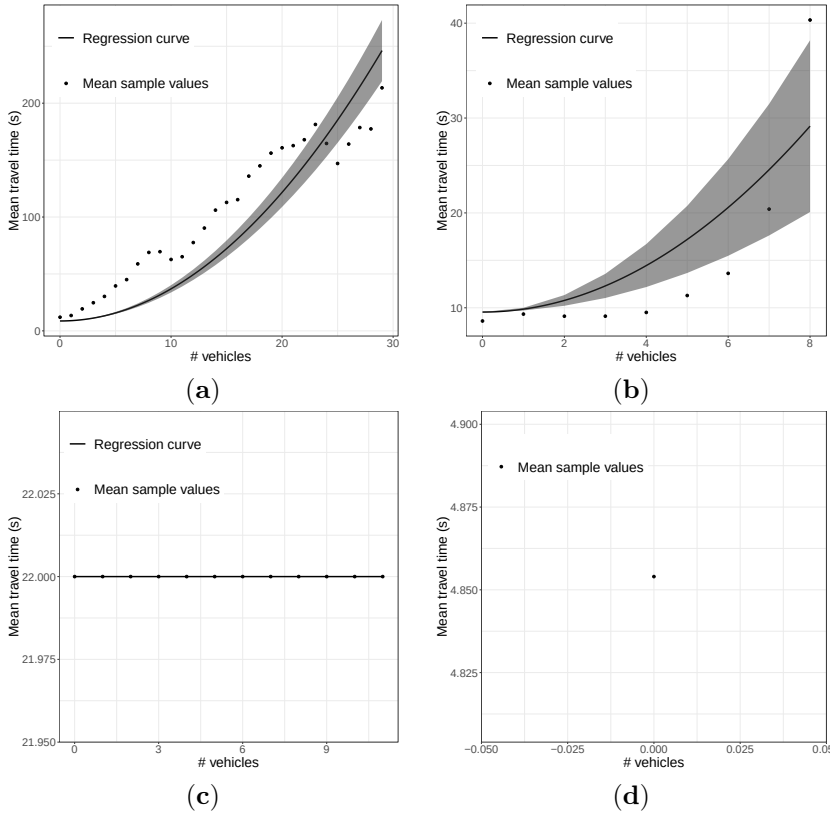


Figure 6.4: Segment Classification with uniform traffic load regulation: (**a**) increasing; (**b**) behavior of a segment previously showing a decreasing behavior; (**c**) constant; and (**d**) unique.

variable number of additional vehicles to be injected at each traffic source location. The variation of the total number of vehicles injected in this scenario ranges from 2 271 to 34 065.

The effect of saturating our scenarios with more vehicles is significant, as the prediction of the resulting traffic through the quadratic regression now suppresses the second class (decreasing) detected above, being that only three types of behavior remain, as shown in Figure 6.4. The first type, called incremental, agrees with general traffic theory, merely stating that the greater the number of vehicles ahead, the greater becomes the travel time along a street. As shown in Figure 6.4b, segments that previously had a decreasing trend now clearly have an increasing behavior, therefore agreeing with traffic theory. However, the segments belonging to the constant classification continue to persist for the same characteristics (see

Figure 6.4c), as the location of the segment in areas of the periphery of the city, which do not have links with other streets, remain mostly uncongested. In addition, the segments having a unique behavior also persist, as they are in general very small segments only sporadically visited by vehicles (see Figure 6.4d).

A particularity that we have observed in the increasing behavior (see Figure 6.4a,b) when applying the quadratic regression to the segments of the city is the fact that they often fail to adjust properly to the relation of the number of vehicles ahead and the time traveled. This causes a high standard error, meaning that the regression does not adequately represent the actual behavior of the segment. Due to this inconvenience, we considered necessary to find another type of regression that better adjusts to the behavior of the segments of the city to reduce the prediction error.

6.5 Segment behavior characterization with logistic regression

In Section 6.4, when analyzing the behavior of the traffic in a city following general traffic theory criteria, we can observe that the polynomial regression fitting results can be deemed inadequate for most segments. For this reason, our aim is to find an alternative mathematical function that better adapts to the behavior that characterizes the different segments of the city. In this section, we propose using a mathematical function that belongs to the logistic family of functions. In particular, we picked the sigmoid function to represent the growth patterns detected in our data set. Thus, we have a logistic regression to predict the outcome of a variable that can adopt a limited number of categories based on independent or predictor variables, and this kind of regression is used to model the probability of an event that occurs as a function of other factors [36]. To this end, we rely on the simple sigmoid function defined by the following mathematical expression:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{6.1}$$

Since we have to adapt the curve of this function to the travel time in free-flow conditions (zero vehicles ahead), we add a parameter b to the Equation 6.1, together with a second term of the initial function to make this possible. In particular, parameter t_{ff} allows defining such free-flow travel time:

$$f(x) = \frac{1}{1 + e^{b-x}} - \frac{1}{1 + e^{b}} + t_{ff} \tag{6.2}$$

Finally, to be able to adapt Equation 6.2 to meet the actual maximum value for the travel times measured, we extend this equation by adding parameter a,

and determine its corresponding displacement in the axis of the abscissa with the parameter c, as shown in Equation (6.3).

$$f(x) = \frac{a}{1 + e^{b - \frac{x}{c}}} - \frac{a}{1 + e^{b}} + t_{ff} \tag{6.3}$$

Our Traffic Prediction Equation is the one adopted for the regression analysis that follows.

6.6 Validation of the logistic regression under different congestion levels

In the previous section, we have presented a logistic function that better describes the congestion behavior of the segments of our target city. We now study the behavior of the city under different degrees of vehicular congestion. This is achieved by regulating the number of vehicles in our simulation following the same method described in Chapter 5, based on taking the standard number of vehicles during rush hours and injecting an extra number of vehicles in each street segment, and using our Equation 5.1. In particular, we vary the total number of additional vehicles injected into the simulation from 2 271 to 34 065. Notice that the total number of segments for this scenario is 9 859.

Once the simulation results were obtained, the processes described in Section 6.2 & 6.3 with our Algorithms 2 and 3 were performed. The process of characterization and classification of the behavior of the different segments in the city was determined by the relationship between the number of vehicles existing in the segment (abscissa axis) and the average travel time in that segment (ordered axis).

To achieve the characterization and classification of the segments, we performed the logistic regression using Equation 6.3, which gives a better fit than using the second order polynomial function, substantially reducing the mean standard error from 25.6477 to 7.3587, as shown in Figure 6.5. In detail, we found that most of the standard error values for the logistic regression remain below 25 s, and the segments with a standard error greater than 50 s represent less than 1% of the segments of the vehicular network of the city. On the other hand, the standard error associated to the quadratic regression can go beyond 300 s, and the error associated to most street segments is greater than 50 s.

Figure 6.6 shows some examples that illustrate how the logistic regression was able to significantly improve the curve fitting error when compared to the standard approach, based on quadratic regression. Therefore, it becomes clear that, by using a logistic regression to characterize the behavior of the segments, the travel time predictions for different travel loads are significantly improved.

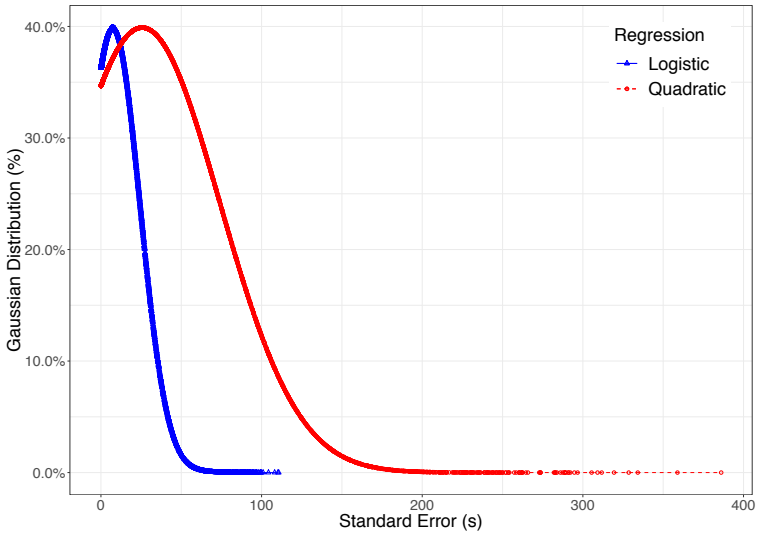


Figure 6.5: Standard error of regressions.

6.7 Clustering results with logistic regression

In the previous section, we highlighted the benefits of adopting a function belonging to the logistic family to properly represent the behavior of the different streets in a city in terms of their travel time characteristic for different traffic loads. Once this step was completed, we then proceeded to perform an appropriate classification of the different streets according to their pattern. Thus, we applied a clustering technique to correctly classify the different groups of streets according to their behavior.

The number of street segments for this scenario is 9 859. The representation of each characterized group was done using a learning machine technique called K-means [37], and we used the parameters of the logistic regression in Equation (6.3) to perform an automatic categorization of the different street segments, assigning each segment to a specific category according to its behavior. With respect to the parameters used as input for the characterization of the segments, we have parameter a, which allows us to discriminate between increasing and constant trends, as it represents the amplitude of the regression curve in the ordinate axis. Likewise, parameter c gives us the characteristic of the maximum extension of the sigmoid curve on the abscissa axis, which represents the number of vehicles in the segment. Another input parameter for the classification procedure is t_{ff}, which represents the travel time on a segment when there are no vehicles ahead, and that helps us to distinguish segments according to their free-flow speeds. Finally, we

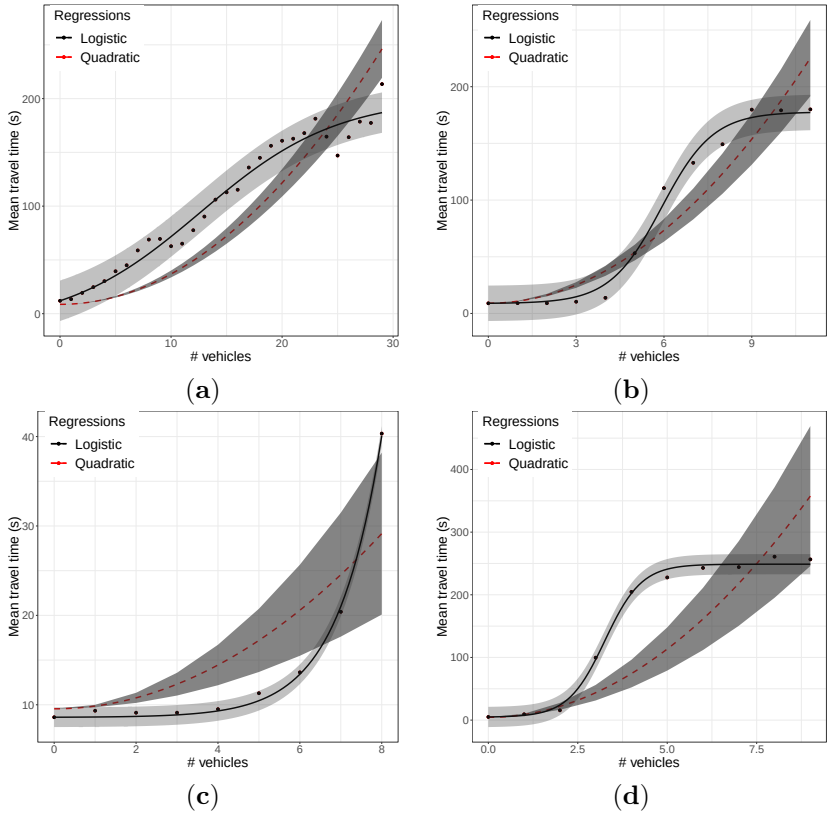


Figure 6.6: Several examples where the logistic regression outperforms the quadratic regression.

.

use a parameter called $f(x)_{max}$ that represents the highest travel time associated with a particular segment.

The K-means clustering method clearly identifies three clusters. We then used the Principal Component Analysis (PCA) [38] procedure to reduce the graphical representation of the four input parameters to two dimensions, and thus be able to understand the classification of the segments in a visual manner. Figure 6.7 shows the result of applying K-means clustering to the city of Valencia. We observe that the percentage of segments with the expected increasing behavior is 81.76%. Likewise, we can observe that the percentage of segments within the constant category is 1.91%, while 16.33% of segments of the city belong to the unique segments family, which are in general quite small segments than remain despite applying our segment reunification algorithm.

In general, the presence of segments characterized by a constant value, even though many vehicles are injected into the segments of the city, can be a problem in the sense that such behavior is not realistic. On the other hand, the unique segments do not reflect the effects of traffic saturation because many of them still represent very small partitions, as in the case of a roundabout that fails to accomplish the conditions of Algorithm 2, impeding several very small segments from being unified. For this reason, a more in-depth analysis of the actual segment lengths has been performed. We found that there were segments having a length that is less than the length of a standard vehicle. Thus, for our study and overall purpose of predicting traffic delays, such segments are useless. In fact, each segment whose size is inferior to, at least, the length of the vehicle plus the inter-vehicle space, can be discarded. To achieve this, we opted for the criterion proposed by Cal y Mayor and Cárdenas [39], which is a theory of vehicular flow that accounts for vehicle flow, speed, density, interval, and the spacing between vehicles. According to these authors, the fundamental equation of the vehicular is able to relate a constant approximate speed, the average free time interval between two vehicles, and their average spacing. Thus, we applied it to our scenario, and the obtained result is very close to 20 m in length for any segment,

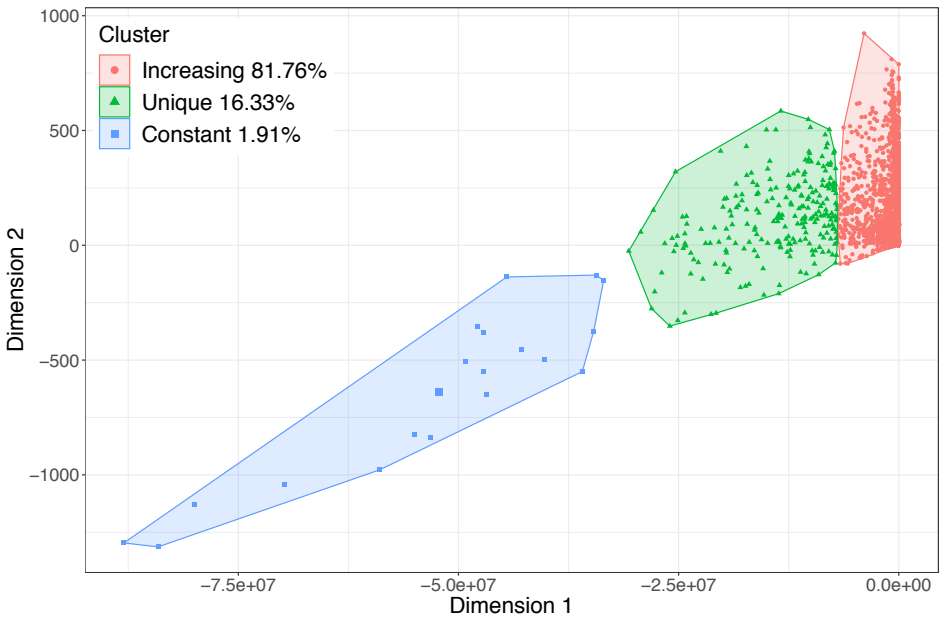


Figure 6.7: Segment classification with logistic regression by clustering.

which gives us a threshold to filter out any segments that measure less than 20 m. Figure 6.8 shows that segment lengths of the different categories follow a Gaussian distribution, and we can visually perceive the effect of such filtering threshold. In fact, a high percentage of segments belonging to the unique family are below this threshold. With respect to the other two categories, only a small percentage of these segments fall below the threshold. Thus, we consider it to be adequate for our purposes.

After filtering out these small street segments, we proceed to identify what is now the actual percentage of segments that belong to each category by again applying the clustering algorithm, and retrieving its corresponding visual representation through the PCA procedure. Figure 6.9 shows that the percentage of segments in the first category now grows up to 92.03%, and that the percentage of segments in the unique category drops to only 6.81%, which is a significant decrease. Finally, a similar number of street segments (1.15%) remains in the constant behavior category.

Figure 6.10 provides an overview of the actual geographical distribution of the streets segments belonging to each of these categories in the city of Valencia. As expected, all major city arteries belong to the "increasing" category, being that only very small and remote segments, located in secondary streets, belong to the two other categories.

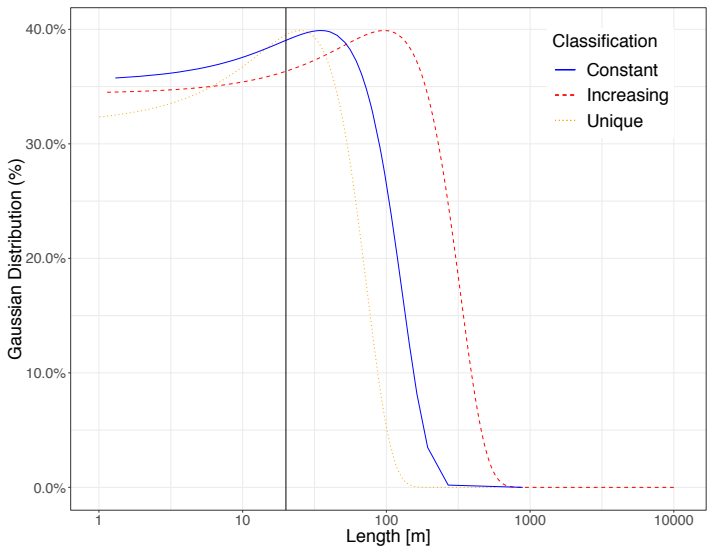


Figure 6.8: Normal distribution of segments length.

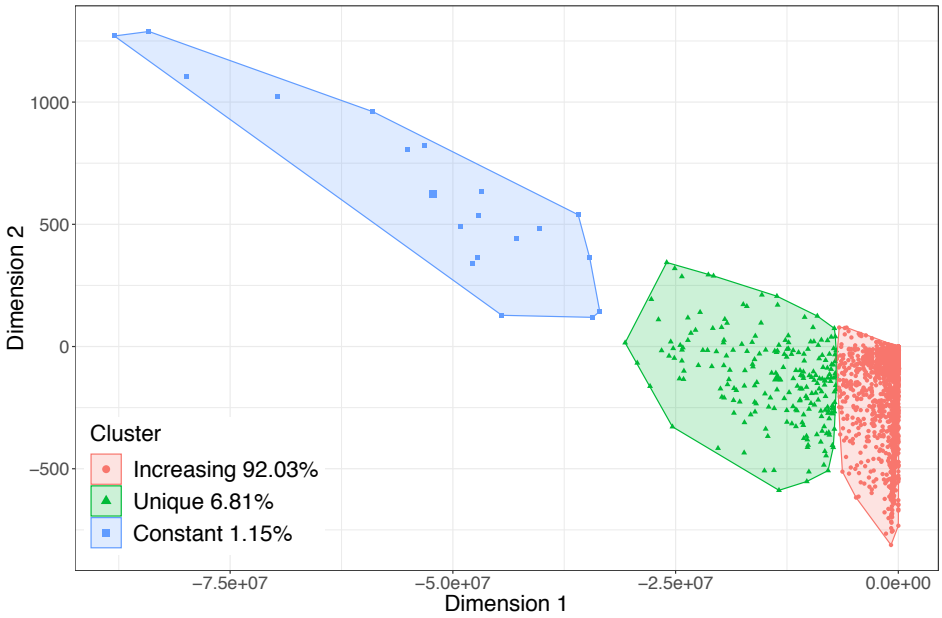


Figure 6.9: Segment classification through clustering for the logistic regression, after applying the filtering threshold.

6.8 Hotspot-based traffic congestion behavior

In this section, we focus on situations where a public event reaches a high number of people for a restricted area, causing the city to experience a heterogeneous congestion effect. For this scenario, which we named "hotspot", we have chosen an area of 270 m radius centered on the Mestalla football stadium, and that has 106 different predicted routes passing nearby. The strategy to achieve our goal is to gradually inject vehicles in this scenario from 100 to 10 000, which are scattered throughout an area including a total of 887 segments. Thus, this scenario combines both "regular vehicles", just passing by that area as in a regular day (departing and arriving), and "hotspot vehicles", which depart from the hotspot area and move to any random destination. Our goal is to study the effects of such asymmetric congestion states when compared to the situation studied earlier, where congestion is more homogeneous. In particular, we want to show if such localized traffic congestion can generate conditions that allow performing a better characterization of the different street segments in terms of their associated travel times prediction curve.

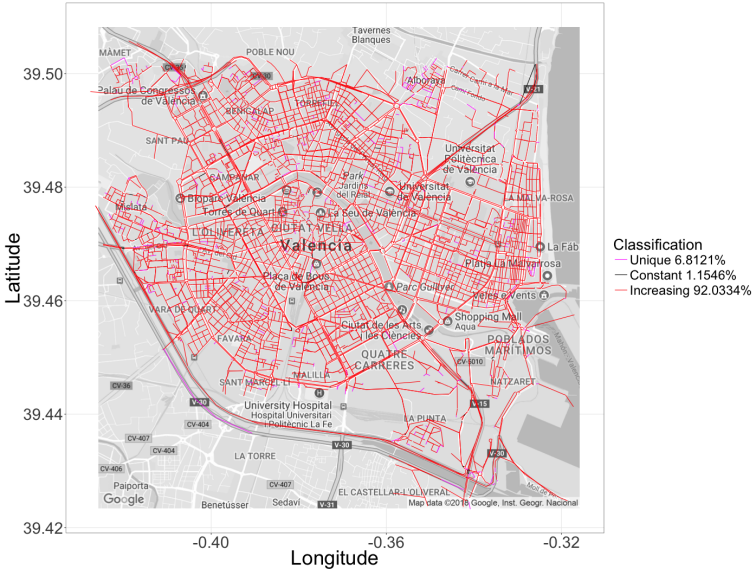


Figure 6.10: Geographical distribution of the segments classification.

For this scenario, we applied Algorithms 2 and 3 to the segments that belong to the hotspot, and obtained the travel times prediction. Afterwards, we applied the logistic regression (see Equation 6.3), and obtained the behavior of the different street segments. Again, we found that the behaviors of the segments within this area belong to the same three categories reported before: increasing trend, unique value, and constant value. As explained previously, the presence of micro-segments in these scenarios persist as they cannot be reunified since they fail to meet the four conditions presented in Section 6.2. Thus, we again applied a filter to the length of these segments to discard excessively tiny ones that are irrelevant in the scope of our case study. We then proceeded to perform the automatic classification through the clustering algorithm together with the PCA, to visualize those groups in a two-dimensional space. As shown in Figure 6.11, the clustering technique now shows that 97.21% of the street segments belong to the main category, meaning that the majority of street segments can now be properly characterized in terms of travel time behavior for different vehicle congestion levels. The two remaining groups are now associated to a very low percentage of segments: 1.7621% for a the unique value category, and 1.0279% for the constant time case. Likewise, we can observe in Figure 6.12 the geographical location of the segments within the studied scenario, differentiated according to their behavior. We can see that the segments for which we have a poor delay characterization (unique/constant value cases) are indeed not quite relevant from a global perspective, being that for the

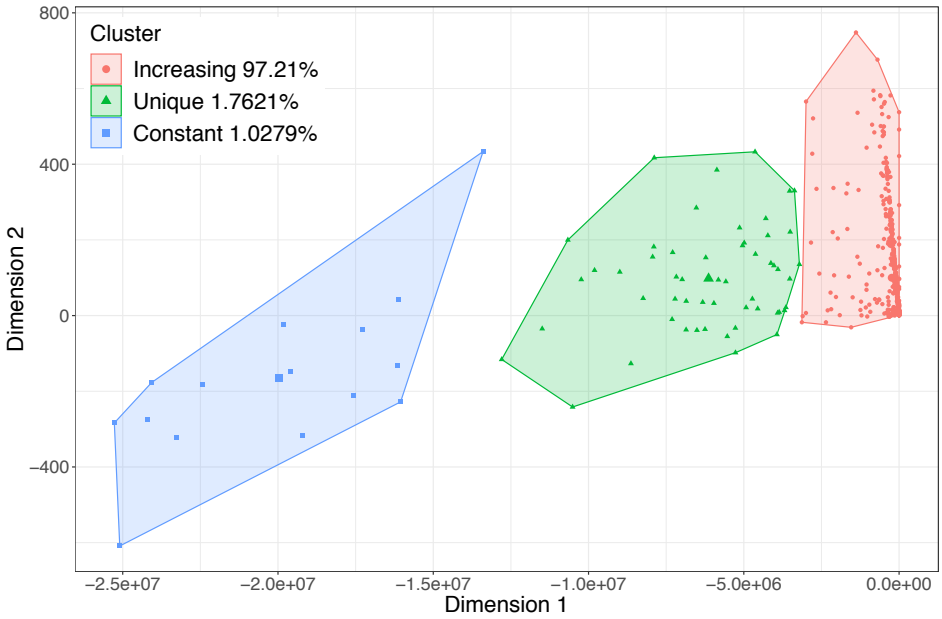


Figure 6.11: Segments classification through clustering for the logistic regression in the hotspot scenario.

majority a clear view of the travel behavior can be obtained, and such per-segment characterization used as input to a larger route planning system.

6.9 Summary

In this chapter, we have developed a methodology to characterize and an algorithm to predict vehicle travel times for the different street segments of a city, under variable degrees of congestion. Since our technique improves travel time estimates, it enables a better control over traffic congestion.

An analysis of the city street model showed that it was necessary to perform a post-processing of the street graph so as to merge segments when excessive fragmentation was detected. Then, we sampled the travel times of vehicles entering a segment, along with the number of vehicles already in that segment, to extract a relation between street occupation and travel time for each street segment. Through regression analysis we showed that a quadratic fit was inadequate in many cases, as the adjustment between the number of vehicles ahead and the time travelled was poor. Thus, as an alternative, we proposed a logistic function,

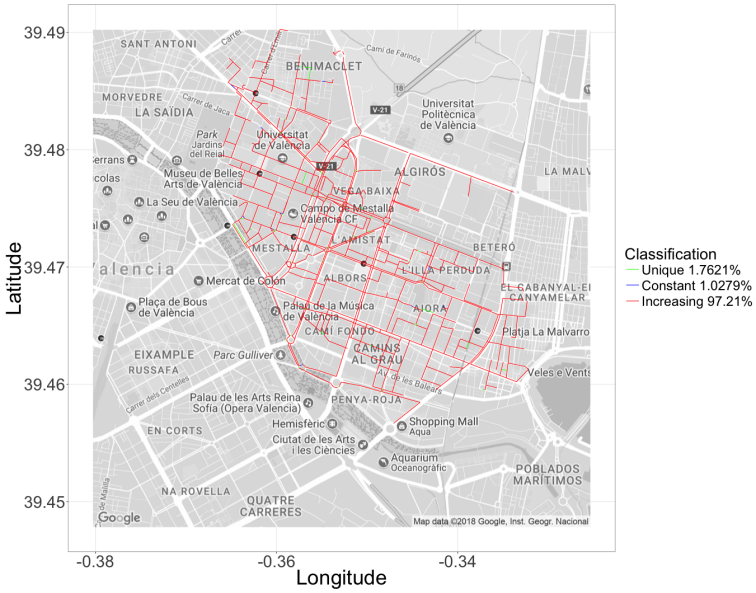


Figure 6.12: Geographical distribution of the segments in the different clusters for the hotspot scenario.

which has proved to provide much better fitting results. In particular, we validated it by injecting an extra number of vehicles in each street segment in the traffic simulator, and checking if prediction accuracy was maintained.

Based on our regression results, we proceeded to perform an appropriate classification of the different streets according to their pattern. Thus, we applied a clustering technique of the different street segments, showing that they can be classified into three well-defined categories, which evidences a predictable traffic distribution using the logistic regression for this city during rush hours.

Chapter 7

Proposed traffic balancing scheme

7.1 Introduction

In this chapter, we describe the procedure followed to improve the traffic flow in the city of Valencia, Spain. As we have mentioned before, the main objective of this thesis is to propose and implement a centralized route manager for autonomous vehicles with the ability to optimize and balance traffic flows by accounting for present and future traffic congestion conditions. Figure 7.1 shows the proposed architecture of our solution. To achieve our goal, we have divided our development into three phases.

The first phase was to generate an O-D matrix for traffic that resembles the real traffic distribution, and that can be directly imported by a traffic simulator through an iterative heuristic for improving traffic congestion modeling. This is clearly explained with their experiments and results in Chapter 4 and 5.

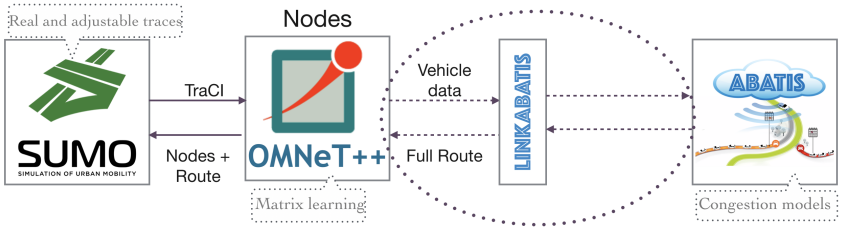


Figure 7.1: Centralized route management architecture.

Having an O-D matrix that resembles the real traffic as input for our traffic

simulator allows us to have meaningful information about the traffic conditions, such as number of vehicles, travel time, and number of vehicles in each segment ID throughout the simulation time. In the second phase, called *Matrix learning*, we basically used all available traffic information to create curves describing traffic behavior in terms of travel time under different loads. For this reason, in this phase we developed the Traffic Prediction Equation 6.3 to characterize travel times over a segment. In this equation ($f(x) = \frac{a}{1+e^{b-\frac{x}{c}}} - \frac{a}{1+e^{b}} + t_{ff}$), variable x represents the number of vehicles in the street segment, t_{ff} is the free-flow travel time in the segment, b is a parameter to adapt the curve of the formula to the travel time in free-flow conditions (zero vehicles ahead), a is the actual maximum value for the travel times measured, and parameter c determines its corresponding displacement in the abscissa axis. Also, the function used belongs to the sigmoid family; in particular, we performed logistic regression to achieve an accurate curve fitting of results for most of the street segments under analysis, as we can observe in Figure 7.2.

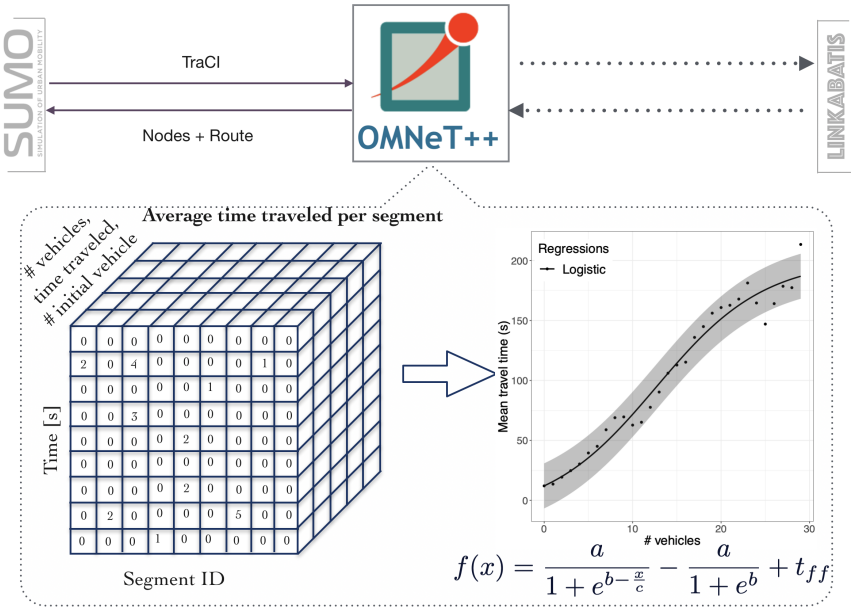


Figure 7.2: Matrix Learning Phase.

Once the traffic behavior was obtained through the Traffic Prediction Equation using the logistic regression, which is able to adequately predict travel times depending on the degree of traffic congestion on a per-segment basis, as explained in the Section 6.5, this equation is integrated into our connection interface to perform the calculations required to predict traffic behavior, obtaining the time

taken for a vehicle to cross a street segment. Thus, we proceed to the third phase called *Congestion models* (see Figure 7.3), which will be described in this chapter, for predicting traffic distribution and the optimization of the routes taken by autonomous vehicles throughout the city.

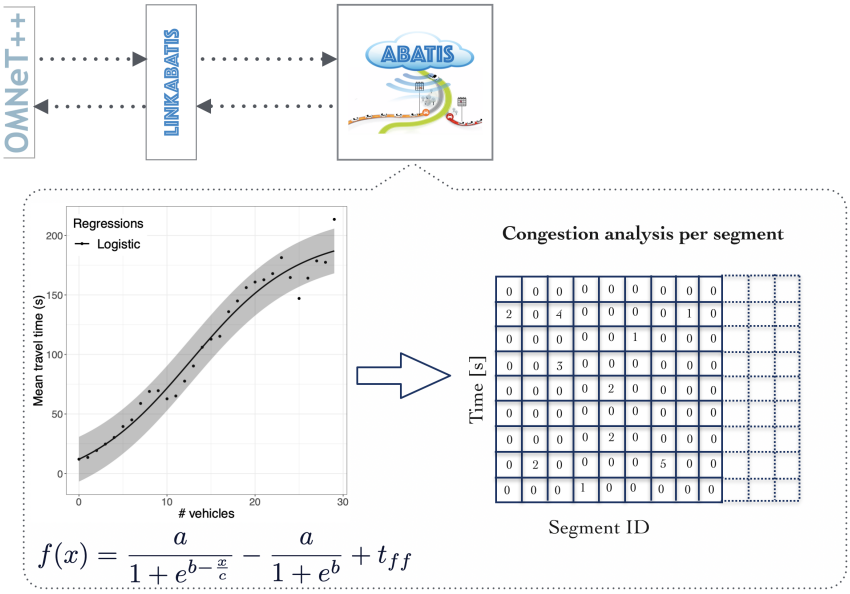


Figure 7.3: Matrix Learned Phase.

In particular, our objective is to demonstrate the validity of the proposed characterization and traffic prediction for this city. With this purpose in mind, we will compare the original traffic conditions with our proposed approach, which is able to balance traffic by exploiting different alternate routes to decongest traffic in critical areas. To achieve this goal, we have connected our vehicular simulators (SUMO and OMNeT++) with the ABATIS route server. However, since these simulators and the route server use different formats for geopositioning data, they cannot communicate directly. Hence, we developed an interface that performs the necessary calculations to perform a conversion between both formats. So, we use our route server capable of handling all the traffic in a city and balancing traffic flows considering present and future traffic congestion conditions. Notice that the limitations of the proposed centralized architecture are closely associated to the characteristics of the server used for that task. In our experiments, we have used a PC with modest features - Intel Core i5 quad-core processor with 16GB of RAM - finding that it is possible to serve up to 2 000 vehicles per second without using parallel processing, and with an average CPU consumption of only 10%.

Also, each response from our server takes between 0.5 ms to 4 ms, depending on the length of the route requested by the vehicle. Thus, we do not foresee any scalability problems for our solution. Therefore, the final solution for centralized route management of autonomous vehicles will be detailed below. In the following section we describe in more detail how this interface works.

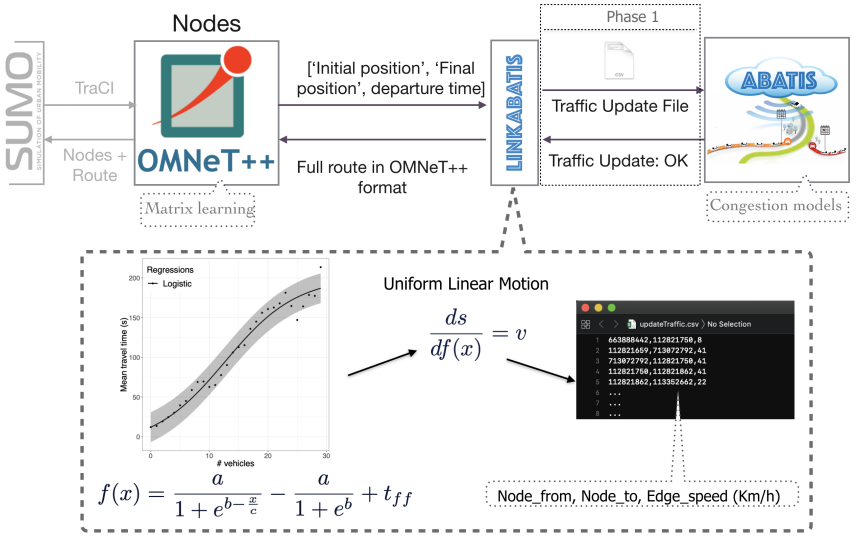
7.2 linkABATIS smart interface

The existing interface between the SUMO traffic simulator and the OMNeT++ network simulator is called TraCI [19], and it allows us to investigate how vehicles move within urban areas when having various traffic patterns, and using real-life driving models. The TraCI interface connecting SUMO and OMNeT++ is open source, and it allows having full control of the vehicles' mobility, and of the attributes of each simulated vehicle. We modified this architecture to obtain more information about simulated vehicles, such as the segments that a vehicle has passed, the distance traveled by a vehicle, the average speed per segment, and for how long a vehicle has been in each segment. We then integrated such information within our process for characterizing and predicting the traffic in the city of Valencia.

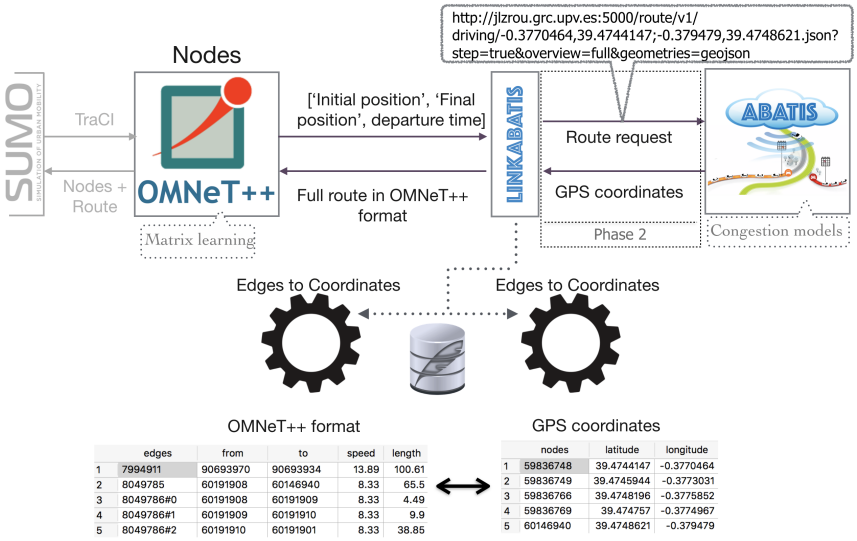
Due to the need to have a route server that offers the routes to the vehicles with a periodic update of the traffic through the vehicular behavior previously performed, we have linked the simulators (SUMO & OMNeT++) to route server (ABATIS) with our interface called *linkABATIS*. This interface allows bidirectional communication between the simulators and the route server with the purpose of offering the best routes that are requested by the simulated vehicles. On account of the format differences between both systems, ABATIS, as a route provider, is not 100% accurate; for instance, if we have a long street within a route, this route server would choose to deliver the geographical coordinates associated with the beginning and end of the street, but not its intermediate points, which are at times required. For this reason, it is necessary to make certain corrections so that the information sent by ABATIS can be understood and processed by OMNeT++ and SUMO. To this aim, we perform searches among the Edge IDs, and match the missing Edge ID through the identification code of the nodes, and this way we are able to correct the gaps in the route to be used by the simulators. Thus, in Algorithm 4 we detail the process followed when a vehicle requests a new route; notice that it updates its knowledge base so as to know the conditions of future traffic. This becomes possible through two phases that follow each request, as we will see below.

7.2.1 Traffic update phase

One of the objectives to be achieved when developing the linkABATIS interface was to solve the traffic congestion problems detected in some parts of the city in



(a) Traffic update phase.



	edges	from	to	speed	length
1	7994911	90693970	90693934	13.89	100.61
2	8049785	60191908	60146940	8.33	65.5
3	8049786#0	60191908	60191909	8.33	4.49
4	8049786#1	60191909	60191910	8.33	9.9
5	8049786#2	60191910	60191901	8.33	38.85

	nodes	latitude	longitude
1	59836748	39.4744147	-0.3770464
2	59836749	39.4745944	-0.3773031
3	59836766	39.4748196	-0.3775852
4	59836769	39.474757	-0.3774967
5	60146940	39.4748621	-0.379479

(b) Route request phase.

Figure 7.4: Phases of our smart Interface

the best way possible. Specifically, it will enable vehicles to use alternative routes to balance the traffic, thus improving traffic flow during rush hours.

Algorithm 4 Traffic update and optimal route request.

Require: Road Network, Vehicle files, Edge files, Reunified Segment file, Segment-info files, and Statistical learning by segment file
Ensure: Traffic Update file, Optimal Route
1: firstTimeInABATIS = **true**
2: **while** ABATIS server route is activated **do**
3: **if** firstTimeInABATIS **is true then**
4: *Extract module* ← extract a graph out of the OpenStreetMap base map
5: *Partition module* ← partition the graph recursively into cells
6: *Customize module* ← customize the cells with default weights for all cells
7: *Datastore module* ← load the data into shared memory for sharing among different processes (requests)
8: *Routed module* ← Run the route server engine that will hot-swap to the new dataset
9: firstTimeInABATIS = **false**
10: **else**
11: **for** edge **in** Edges file **do**
12: $edgeIDInitial[]$ ← vector that stores all initial edge IDs where the vehicles will be injected into the road network as starting points
13: **end for**
14: **for** vehicles **in** Vehicles file **do**
15: **if** $(firstEdge\_Vehicle = edgeIDInitial[])$ **then**
16: $vehicle\_info[]$ ← vector that stores all information about vehicles together with the time that the vehicle will be injected into the road network
17: **end if**
18: **end for**
19: **for** vehicles **in** Statistical learning by segment file **do**
20: **if** $(vehicle[ID] = vehicle\_info[vehicleID])$ **then**
21: $\nu[]$ ← vector that stores all numbers of vehicles in the segment before a new vehicle enters the segment
22: **end if**
23: **end for**
24: **for** segment **in** Statistical learning by segment file **do**
25: **if** ((vehicle_info[segmentID] = segment[ID]) **and** (vehicle_info[segmentID] **not** partitioned)) **then**
26: v = segment[speed] ← gets the same original speed of the non-reunified segment
27: **else**
28: a, b, c=segment[ID][a][b][c] ← get the three parameters that characterize the traffic using the logistic function
29: t_{ff}=segment[ID][t_ff] ← get the free-flow travel time (zero vehicles ahead) in the segment
30: $f(\nu) = \frac{a}{1+e^{b-\frac{\nu}{c}}} - \frac{a}{1+e^{b}} + t_{ff}$ ← get the mean travel time through the Traffic Prediction Equation knowing numbers of vehicles ahead
31: v=$\frac{ds}{df(\nu)}$ ← get the speed through the Linear Motion Equation knowing the mean travel time and the segment length
32: **end if**
33: Traffic Update file ← stores all nodes that compose the edges that will change their weight in the routes server together with their new speed (v)
34: **end for**
35: ABATIS server route ← receives the Traffic Update file that specifies edge weight updates
36: *Customize module* ← customize the cells by calculating routing weights for all cells through Traffic Update File
37: *Datastore module* ← load the data updated into shared memory for sharing among number of processes (requests)
38: linkABATIS → transform initial and final position from simulators format to GPS coordinates
39: ABATIS server route ← receives vehicle information including ID, initial and final position in GPS
40: *Routed module* → return the optimal route according to the traffic updated in GPS using MLD Algorithm
41: linkABATIS → transforms the optimal route from GPS to simulators format for $vehicle[ID]$
42: **end if**
43: **end while**

To accomplish this goal, it is necessary to have an individual characterization of street segments in terms of average travel times experienced by vehicles for different degrees of congestion, whose estimation is based on the number of vehicles found ahead of a vehicle that just enters a segment. Then, the linkABATIS interface processes and sends the necessary information to the ABATIS route server so that it updates the weights of the street segments by accounting for the expected segment occupation to estimate speed levels. The speeds for each segment are calculated using the Uniform Linear Motion equation ($\frac{ds}{df(x)} = v$). Notice that, beforehand, we have the average travel time estimation, as provided by Equation 6.3, along with the length of each street in the city.

The scheme of the first phase of our intelligent interface, is shown in Figure 7.4a.

7.2.2 Route request phase

In a second phase, the linkABATIS interface is used to query for the route of a vehicle accounting for the updated traffic. In order to send this query to ABATIS, it is necessary to know the initial geographic location and the target geographic location of the vehicle, together with the vehicle time of departure. Based on this input, ABATIS will return the best complete route as a sequence of geographic coordinates. However, the format of the route proposed by ABATIS is not compatible with the format accepted by SUMO and OMNeT++. To address this incompatibility, it is necessary to create a translator function that converts the geographical coordinates into codes that are compatible with both the network and the traffic simulator, as well as the inverse conversion to support bidirectional communications. To improve the speed of processing and search in the conversion of geographic coordinates from one format to the other, we store the information relevant to all nodes, such as their ID code and geographic coordinates. Also, we store the "edges", which are representations of streets or roads in the context of SUMO. These edges contain the identification code of the corresponding nodes, edge type information, the edge length, the vehicular speed associated to that edge, and the different street segments that have been unified, as presented in Chapter 6.

Figure 7.4b represents the scheme of the second phase of our smart interface.

7.3 Uniform traffic load balancing

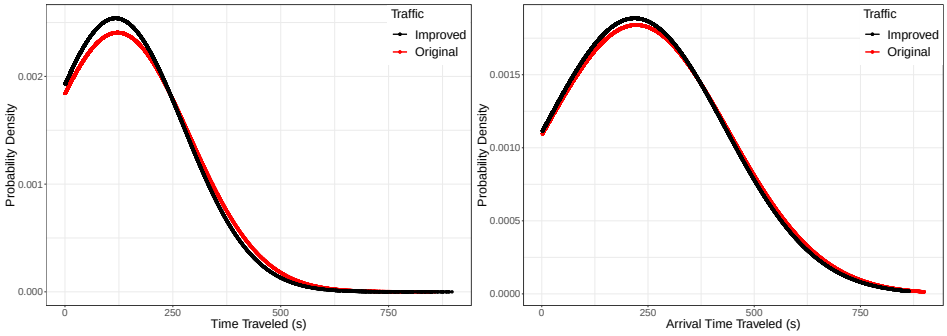
In this section, we study the impact of load balancing traffic throughout the whole city of Valencia (excluding suburban areas), having a size of 77.42 km^2. To achieve this goal, the reference traffic scenario for the city of Valencia consists in injecting a total of 34,065 vehicles corresponding to real traces at the rush hour, as mentioned in Chapter 5, and for a period of 900 seconds. This scenario allows us to check

the effectiveness of our traffic congestion equation for a city, reducing congestion, and balancing the flow of traffic globally.

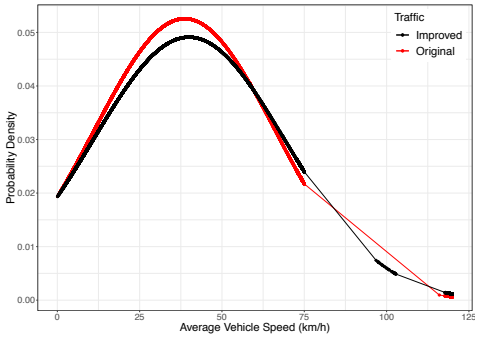
Figure 7.5a shows the improvements that have been obtained in terms of average travel times for the entire city, evidencing that our traffic prediction was effective. In particular, we find that the prediction of traffic reduces the time the vehicles require to arrive to their destinations. We can also see that the average number of vehicles traveling on the streets, and falling within the Gauss Bell, is approximately 5%, and there are very few vehicles whose travel time is high compared to the vehicles in the reference traffic.

Likewise, in Figure 7.5b we observe that vehicles tend to arrive much earlier when our load balancing technique is used compared to the normal behavior of the reference traffic; in fact, the standard deviation is reduced even taking into account that the scenario encompasses the whole city.

Another metric used to assess vehicular traffic behavior is the average speed that vehicles have in each of their routes. Thus, in Figure 7.5c we can observe



(a) Average Travel Time. (b) Arrival times for the vehicles.



(c) Average vehicle speed.

Figure 7.5: Improvements our proposed approach in the city of Valencia.

that, when doing traffic load balancing, vehicles move faster than under reference traffic conditions. This is mainly due to the fact that, when applying our strategy, a vehicle has several routes to reach its destination, and the less congested route is chosen for the vehicle to arrive at its destination.

Likewise, it can be observed that, in the improved traffic results, our algorithm is able to reduce congestion in some of the streets, allowing vehicles to increase their speed progressively and efficiently, verifying the correct balancing of traffic

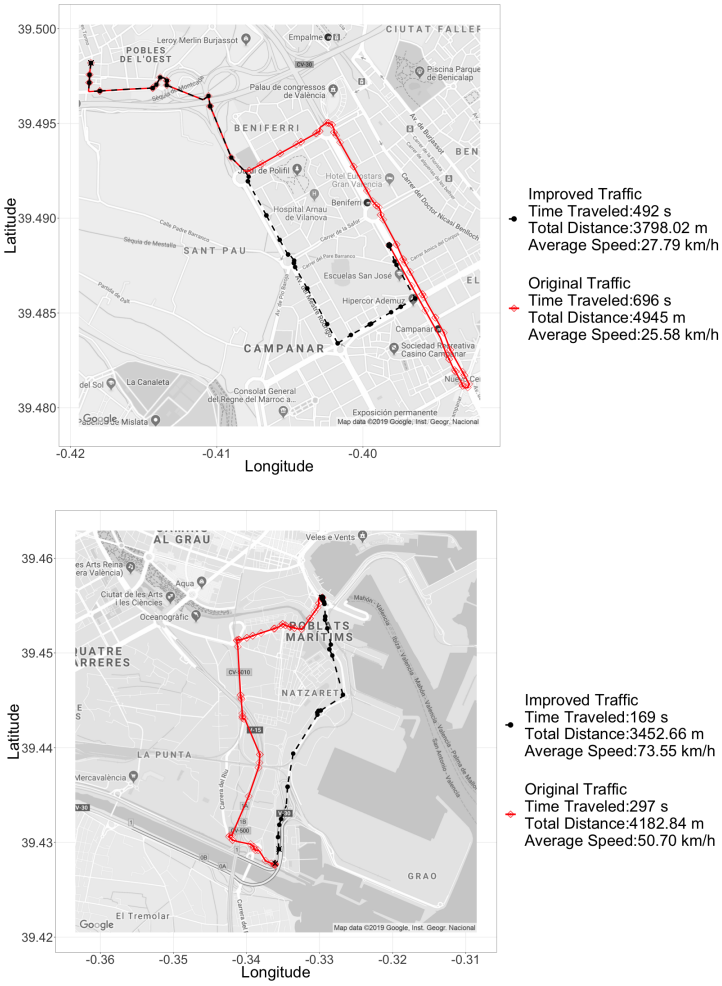


Figure 7.6: Examples of long routes in the city of Valencia.

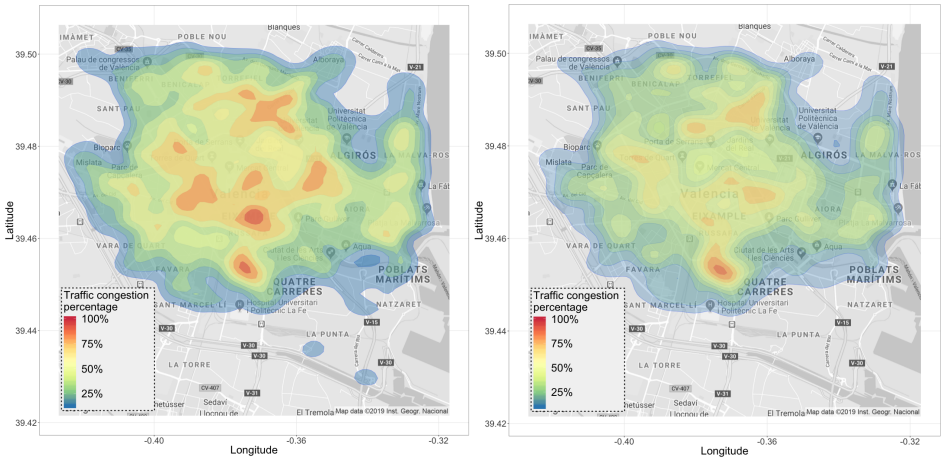
introduced by our algorithm.

To compare the routes taken by the vehicles within the reference traffic scenario against the scenario with a balanced load, we have chosen some examples of the longest routes, along with their main metrics, for comparison purposes. As shown in Figure 7.6, the route chosen by our algorithm is the optimal between the different route options, introducing a lower travel time, and being less congested at the time the vehicle reaches each of the street segments along the proposed path.

Figure 7.7 provides a broader perspective of the areas that tend to present traffic congestion in the city of Valencia, with a tendency towards the red color. Thus, we observe that the reference traffic in Figure 7.7a has more congested areas; by applying our traffic load balancing algorithm, the amount of congested areas is reduced, greatly alleviating traffic, as shown in Figure 7.7b.

7.4 Hostpot-based traffic load balancing

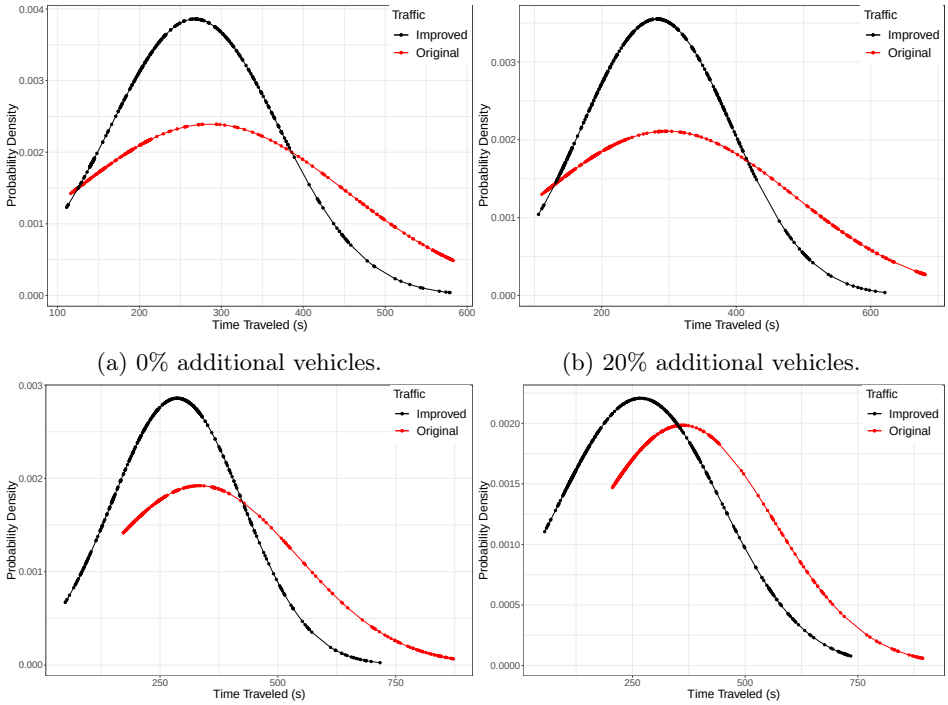
For our next set of experiments, we have chosen the Ruzafa neighborhood within the city of Valencia, whose area is 2 150 × 500 meters, with 300 vehicles moving around during the rush hour. The real trace corresponds to a Monday in November (8:00 am - 9:00 am), as described in Chapter 4, within a maximum period of 900 seconds, and whose O-D matrix is within the selected zone. This particular neighborhood offers several alternative routes to gain further insight into the effectiveness of our architecture, and in particular of our Traffic Congestion Equation



(a) Original Traffic.

(b) Improved Traffic.

Figure 7.7: Heatmap of traffic congestion for the city of Valencia.



(a) 0% additional vehicles.

(b) 20% additional vehicles.

(c) 80% additional vehicles.

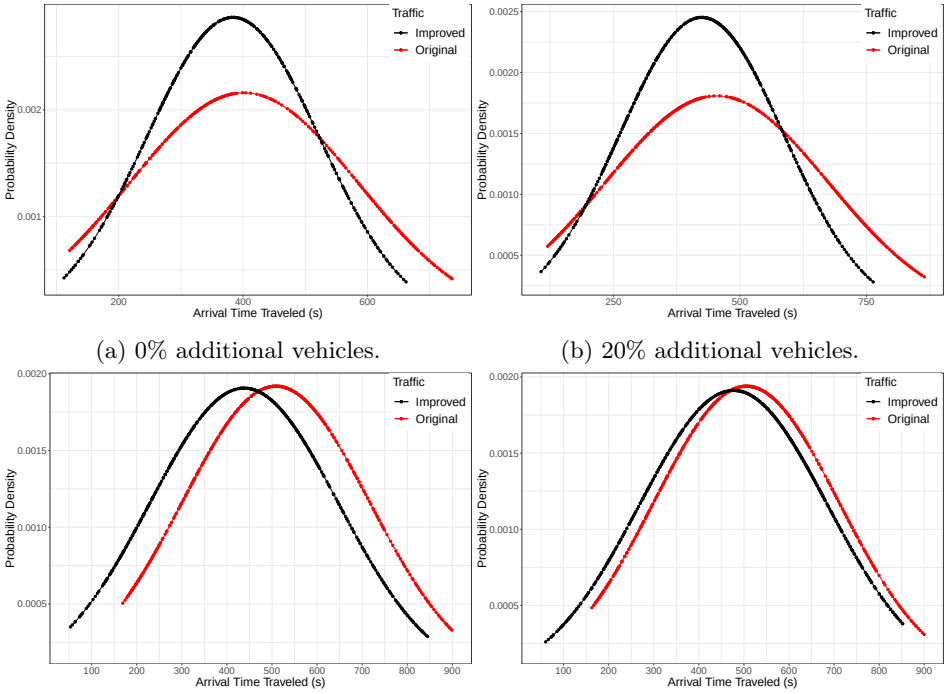
(d) 100% additional vehicles.

Figure 7.8: Average Travel Time under different saturation levels.

6.3, when used to balance traffic flows and reduce congestion. In addition, we have performed tests by varying the total number of additional vehicles injected in this scenario up to twice the number of vehicles of the real trace. For this purpose we relied on Equation 5.1, presented in Chapter 5, thus ensuring that the additional load is distributed between the different street segments involved, and we assess the behaviour for different numbers of additional vehicles.

In Equation 5.1, σ_s is the smallest number of vehicles detected among all segments of the street s, variable ω_s is the number of segments that compose street s, φ_n is the traffic flow adjustment factor for each street segment, and n is the number of iterations. In addition, the term λ_s is used to regulate the number of vehicles injected into the scenario to generate different degrees of congestion, and thus offers the required flexibility to regulate congestion in the target area according to any criteria.

Figure 7.8 shows the improvements achieved in terms of vehicle travel times. We can see that, indeed, our traffic predictions are effective, and that they allow us to significantly reduce the time required for vehicles to reach their destinations.



(a) 0% additional vehicles.

(b) 20% additional vehicles.

(c) 80% additional vehicles

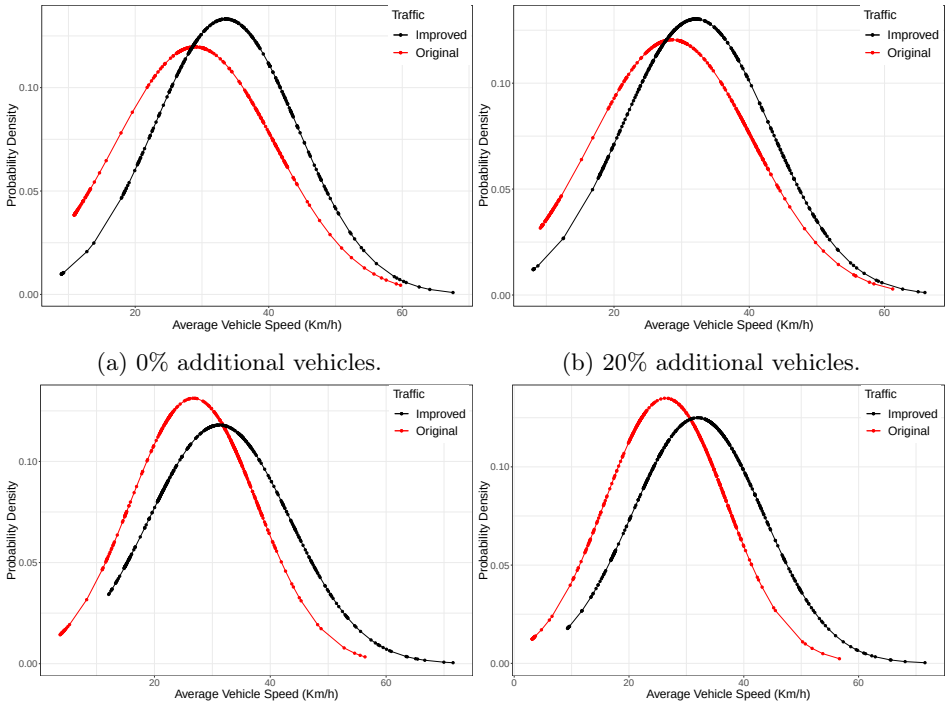
(d) 100% additional vehicles.

Figure 7.9: Arrival times for the vehicles under different saturation levels.

In particular, we observe that the average number of vehicles traveling on the streets within this area falls within a Gauss bell, and that the average travel time improvement achieved is approximately 8% compared to the default travel time behavior (no load balancing).

Likewise, we can observe in Figure 7.9 that the time of arrival to the destination for the vehicles that benefit from our load balancing technique is also decreased when compared to the original traffic used as reference, and the standard deviation for this metric is also reduced.

Another parameter providing insight on how the traffic flow is improved is the average vehicle speed within the scenario. Figure 7.10 shows that vehicles are able to drive faster in the scenario where our load balancing strategy is active. In particular, we observe that the normal distribution characterizing vehicular speed is shifted to the right, which indicates that the average speed of the original traffic is slower. Notice that lower vehicle speeds are closely related to higher traffic congestion conditions, impeding vehicles to advance. Therefore, we find that our improved solution achieves 16% more fluid traffic conditions compared



(a) 0% additional vehicles.

(b) 20% additional vehicles.

(c) 80% additional vehicles.

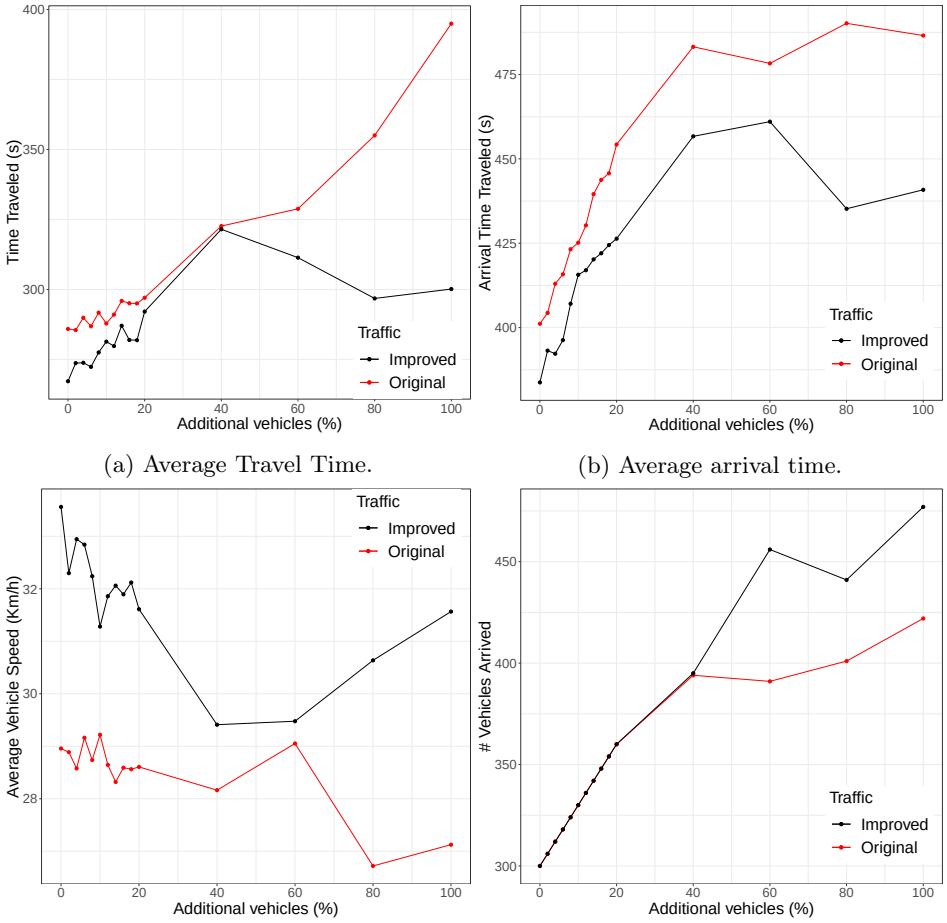
(d) 100% additional vehicles.

Figure 7.10: Average vehicle speed under different saturation levels.

to the default, non-balanced situation.

In Figure 7.11, we can also observe the general statistics when gradually increasing the vehicles in the Ruzafa area. Therefore, we observe in Figure 7.11a how the average travel time increases when injecting additional vehicles. However, when applying our algorithm, travel times do not increase as abruptly as in the reference scenario, meaning that our algorithm assigns vehicles the best routes, achieving traffic balancing at all times. Consequently, we also observe the same behavior in Figure 7.11b, since the vehicles injected in the traffic scenario that benefits from our from load balancing strategy reach their destination faster than the vehicles in the reference traffic scenario.

In terms of the average vehicle speed when injecting additional vehicles, Figure 7.11c shows that the speed of the vehicles is reduced, meaning that there are areas where the presence of many vehicles provokes congestion, reducing traffic fluidity. However, when traffic is managed by our route server, speed does not decrease as abruptly as in the default case, evidencing the effectiveness of our algorithm at regulating traffic, especially in the presence of automated vehicles, as it is



(a) Average Travel Time.

(b) Average arrival time.

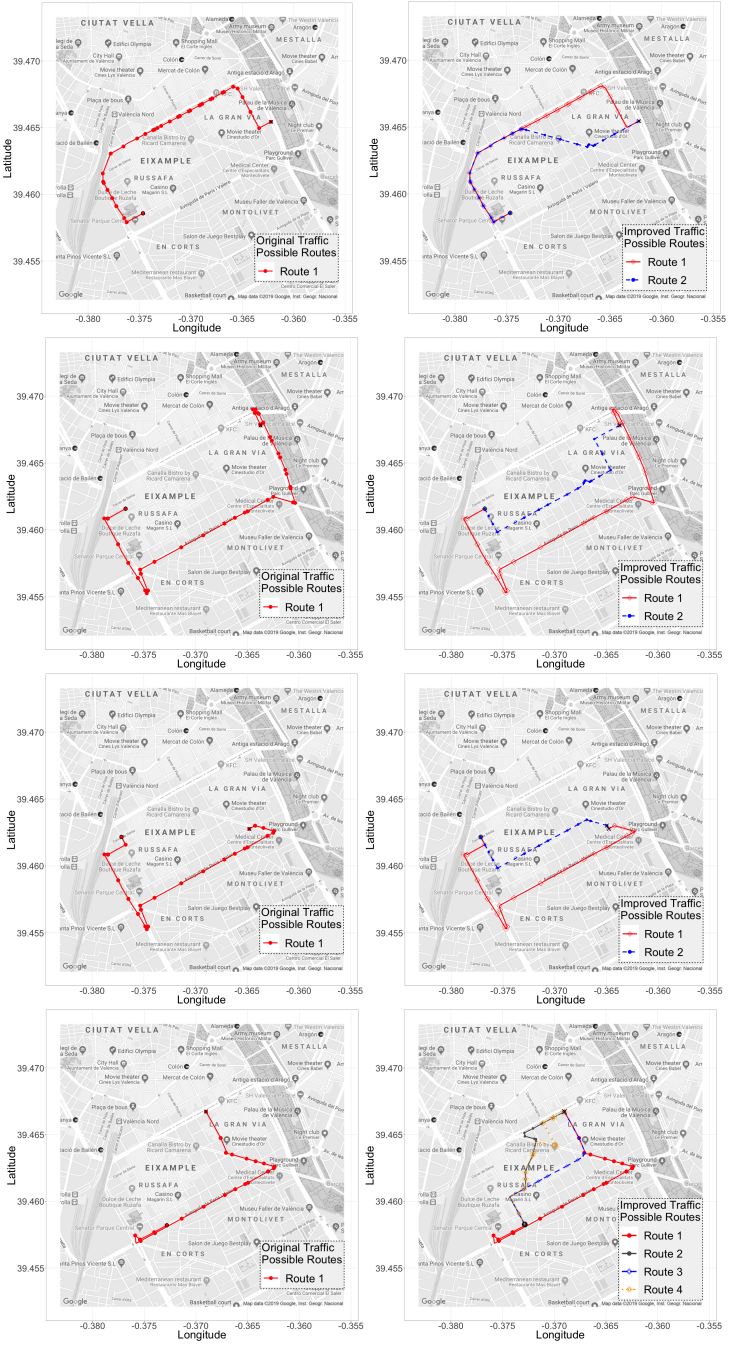
(c) Average vehicle speed.

(d) # vehicles arrived.

Figure 7.11: General Statistics at the Ruzafa neighborhood.

able to reduce the areas experiencing traffic congestion. This is also reflected by the number of vehicles able to fully complete their routes, as shown in Figure 7.11d. By applying our load balancing algorithm to traffic, more vehicles arrive at their destination because they are assigned alternate routes with less traffic, thereby increasing the chances of completing their route within the simulation period tested.

As stated earlier, our algorithm is able to analyze different alternative routes to balance traffic, choosing the least congested route through the prediction of travel times depending on the degree of traffic congestion on a per-segment basis by



(a) Original traffic routes (b) Improved traffic routes

Figure 7.12: Routes Origin-Destination of the original traffic and our propose.

(a) Original Traffic.

(b) Improved Traffic.

Figure 7.13: Heatmap of traffic congestion at the Ruzafa neighborhood.

using Equation 6.3. Thus, we observe in Figure 7.12 some examples of routes for this scenario, in which the reference traffic (see Figure 7.12a) always takes a single route to reach the destination without having prior knowledge of the vehicular behavior at a given time, thus causing traffic congestion and increasing travel times, generating an unnecessary increase in terms of fuel consumption. However, when relying on our route server (see Figure 7.12b), our algorithm always offers several alternative routes to balance the traffic by having the knowledge of the predicted congestion conditions. For that reason, it determines the route offering a lower travel time each time an autonomous vehicle requests such information, and such optimal route will be frequently changing depending on updates regarding future traffic levels.

To complete our analysis, Figure 7.13 allows assessing the performance differences between the experiments in terms of heat maps that represent the vehicle density in the selected scenario. Figure 7.13a shows that the original traffic is distributed inefficiently, failing to occupy all the streets in this area. In fact, we can observe that traffic is mostly concentrated in a single area, where a high accumulation of vehicles provokes traffic congestion; this occurs because only a few routes are used to go from origin to destination, as occurs in real-life situations if relying on a standard navigation tool. Figure 7.13b shows that our solution generates a different behavior. Now, vehicles are assigned alternative routes when traffic congestion is predicted in the streets along the assigned route. In fact, we can clearly observe that more alternative routes are used to decongest traffic, improving traffic fluidity, and effectively balancing the traffic throughout the scenario. This way, only ephemeral congestion events take place near semaphores.

7.5 Summary

In this chapter, we presented a route server capable of handling all the traffic of a city, allowing to optimize and balance the flow of traffic by accounting for present and future traffic congestion conditions.

We demonstrated the validity of the proposed solution by comparing performance under the original traffic conditions (reference) with our proposed approach, which is able to balance traffic by exploiting different alternate routes. To achieve this, we connected our simulators with the ABATIS route server through a smart interface named linkABATIS, which is capable of performing traffic prediction calculations, periodically updating the traffic conditions into our route server, and converting different formats for geoposition data used by our simulators and the ABATIS route server. By integrating the Traffic Prediction Equation developed in the previous chapter, we were able to efficiently predict future traffic conditions in a dynamic manner, and achieve load balancing. In particular, experimental results comparing the real traffic flow against our improved solution showed that the latter achieved significant benefits in terms of travel times and average travel speed of vehicles, reducing congestion and improving traffic flow. Notice that our solution was validated in a large-scale scenario, such as Valencia, and where the complexity increases due to the numerous routes that the vehicles requested from a centralized traffic manager. We also focused on specific areas with great congestion, such as the Ruzafa neighbourhood in Valencia, where we evaluated the impact of injecting additional vehicles into this area so as to check the limits of our traffic balancing algorithm, improving average travel speed and travel time compared to the default traffic conditions.

Chapter 8

Conclusions, Publications and Future Work

8.1 Conclusions

In the near future, an increasing number of autonomous vehicles is expected to be deployed in our cities. As we move towards this scenario, new requirements in terms of efficient traffic management emerge. In particular, different studies highlight that congestion problems can become more prevalent, as more vehicles use navigation systems to select their routes. However, real experiments about traffic congestion are complicated because of the high cost and resources required to perform them.

Therefore, simulations are a key tool to analyze and improve traffic in our cities. However, for simulation results to be representative, the traffic flows injected must be realistic, meaning that traffic patterns, defined by an origin/destination (O-D) matrix for the vehicles in a city, should be provided. Valencia is one of the many cities where traffic analysis is of utmost importance, but that nevertheless has no detailed O-D matrix for traffic analysis; in fact, only induction loop measurements for the most relevant streets/avenues are readily available. In this thesis we proposed a heuristic that, by iteratively refining the output provided by the DFROUTER tool, is able to generate an O-D matrix for the traffic in Valencia that attempts to be a good approximation of the real traffic distribution. By comparing the generated results against existing traffic mobility data for the cities of Cologne (Germany) and Bologna (Italy), we found that the traffic flow definitions obtained for Valencia provide realistic results. In particular, we observed a good

traffic dispersion throughout the different streets of the city, meaning that traffic is flowing through a high number of street segments. In addition, we found that there was a clear asymmetry between streets/segments with low and with high traffic levels, as occurs in real situations. Moreover, it can be observed that injecting additional vehicles into the road network gradually increases the travel time and decreases the average speed of the vehicles, showing a mostly linear behavior that allows easily testing under different traffic loads.

Having a realistic traffic model for a specific target city is a key requirement to obtain meaningful simulation results when issues such as traffic density and traffic patterns can have an impact on the conclusions derived from experiments. Achieving such realistic models typically requires describing traffic in terms of O-D matrices. In addition, if aimed at developing advanced traffic management solutions, it becomes further necessary to have a more in-depth understanding of how traffic is distributed in a particular city, which basically requires performing a correct analysis and classification of such traffic. Hence, our starting point was a realistic traffic model for Valencia. It was used to characterize all street segments in the city in terms of travel times when vehicles face different degrees of congestion. To achieve this characterization, we started by processing the map of the target area in order to merge segments of the same street whenever unnecessary fragmentation was detected and could be reversed. Then, we performed simulation experiments using SUMO to retrieve the travel times of vehicles when facing different degrees of traffic saturation on the traveled segment. Finally, using different regression strategies, we performed curve adjustment to obtain an expression that allowed us to characterize these travel times. This way, we developed an equation able to predict traffic congestion by relating travel times with the vehicular load on each street. Once all segments where characterized, our next contribution was to apply clustering to automatically classify segments according to their travel time behavior. In particular, we applied the K-means technique to generate the clusters, followed by a Principal Component Analysis to extract the main clustering features that enable visual representation. The results of the clustering process clearly define three categories: segments with incremental traffic delays (the majority), segments with constant delays (typical loads do not cause congestion), and single value results, corresponding to small segments rarely visited by vehicles. We complemented this study with an analysis of segment lengths to filter out segments that were too small, and so not representative for our traffic analysis. We also showed how the street segment characterization could be improved by causing very high congestion levels in a hotspot scenario, allowing more than 97% of the segments to be characterized adequately.

Accordingly, to solve congestion traffic problems as more vehicles use navigation systems to select their routes, we proposed a centralized traffic manager able to balance the traffic in a city in a seamless and efficient manner. However, by integrating the traffic equation characterizing each particular street segment in our ABATIS route server, we were able to accurately predict future traffic conditions

in a dynamic manner, thus achieving load balancing. Experimental results comparing the default traffic flow against our improved solution showed that the latter achieved significant benefits in terms of travel times and average travel speed of vehicles, reducing congestion and improving traffic fluidity. Our proposed solution was validated in large-scale scenarios, such as the city of Valencia, with an area of 77.43 Km^2, and where the complexity increases due to the numerous routes that the vehicles requested to our centralized traffic manager. Our proposal improved travel times by 5%, and the average travel speed was 5% higher compared to the reference traffic for the entire city. We also focused on areas facing high congestion levels, such as the Ruzafa neighborhood in the city of Valencia, where we evaluated our proposed solution when injecting additional vehicles into this area to check the limits of our traffic balancing algorithm. In particular, our proposed solution was able to improve travel times by 8% compared to the default traffic conditions, even under very high loads. In addition, the average travel speed with our proposed solution was 16% higher, meaning that vehicles arrived faster to their destinations.

8.2 Publications

This section lists the publications that are a result of this thesis, as well as some other collaborations and related works published during this time.

8.2.1 International Journals

- Zambrano-Martinez, J.L.; Calafate, C.T.; Soler, D.; Cano, J.C. Towards realistic urban traffic experiments using DFROUTER: Heuristic, validation and extensions. *Sensors **2017**,* 17(12), 2921, doi:10.3390/s17122921. **I.F. 2017: 2,677; JCR: Q1 Category**.

- Zambrano-Martinez, J.L.; Calafate, C.T.; Soler, D.; Cano, J.C.; Manzoni, P. Modeling and characterization of traffic flows in urban environments. *Sensors **2018**,* 18(7), 2020, doi:10.3390/s18072020. **I.F. 2018: 2,475; JCR: Q1 Category**.

- Zambrano-Martinez, J.L.; Calafate, C.T.; Soler, D.;Lemus-Zúñiga,L.-G.; Cano, J.C.; Manzoni, P; Gayraud, T. A Centralized Route-Management Solution for Autonomous Vehicles in Urban Areas. *Electronics **2019**,* 8(7), 722, doi:10.3390/electronics8070722. **I.F. 2019: 1.764; JCR: Q3 Category**.

8.2.2 International Conferences

- Zambrano, J.L.; Calafate, C.T.; Soler, D.; Cano, J.C.; Manzoni, P. Using real traffic data for its simulation: Procedure and validation. In Proceedings

of the 2016 International IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/ IoP/SmartWorld), Toulouse, France, 18–21 July 2016; pp. 161–170, doi:10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0045. **Core B**.

- Zambrano-Martinez, J.L.; Calafate, C.T.; Soler, D.; Cano, J.C.; Manzoni, P. Analysis and Classification of the Vehicular Traffic Distribution in an Urban Area. In *Ad-hoc, Mobile, and Wireless Networks*; Puliafito, A., Bruneo, D., Distefano, S., Longo F., Eds.; Springer: Cham, Switzerland, 2017; pp. 121–134, doi:10.1007/978-3-319-67910-5_10. **Core B**.

- Zambrano-Martinez,J.L.; Calafate, C.T.; Soler, D. and Cano, J.-C. Towards a centralized route management solution for autonomous vehicles. In: 2019 IEEE Global Communications Conference: Selected Areas in Communications: Internet of Things (Globecom2019 SAC IoT). Waikoloa, USA, Dec. 2019 **Core B**.

8.2.3 National Conferences (Spain)

- Zambrano-Martinez, J. L., Calafate, C. T., Soler, D., Cano, J. C., & Manzoni, P. (2016, September). Simulación de Tráfico Vehicular en base a Trazas Reales. In I Jornadas de Computación Empotrada y Reconfigurable.

- Zambrano-Martinez, J. L., Calafate, C. T., Soler, D., Cano, J. C., & Manzoni, P. (2017, September). Modelado realista del tráfico urbano usando DFROUTER. In II Jornadas de Computación Empotrada y Reconfigurable.

- Zambrano-Martinez, J. L., Calafate, C. T., Soler, D., Cano, J. C., & Manzoni, P. (2018, September). Distribución del tráfico vehicular en áreas urbanas: clasificación y análisis. In III Jornadas de Computación Empotrada y Reconfigurable.

- Zambrano-Martinez, J. L., Calafate, C. T., Soler, D., Cano, J. C., & Manzoni, P. (2019, September). Modelado y caracterización del flujo de tráfico en entornos urbanos. In IV Jornadas de Computación Empotrada y Reconfigurable.

8.3 Awards, Internship and Projects

In September 2019, this thesis won the 2nd prize in the VI edition of the Awards of the European Week of Sustainable Mobility of the Valencian Community.

During the period January 2019 to April 2019, I did my internship in the research group Services and Architectures for Advanced Networks (SARA) at the

Laboratory for Analysis and Architecture of Systems (LAAS) of the French National Centre for Scientific Research (CNRS) in Toulouse - France. This scholarship was partially founded by the *Erasmus Prácticas* program.

In addition, I have participated in the following projects during this thesis period:

- Walkie-Talkie: "Vehicular Communication Systems to Enable Safer, Smarter, and Greener Transportation", which was funded by the Ministerio de Economía y Competitividad, Spain, under Grant TIN2011-27543-C03-01.
- SmartCarPhone: "Toward Seamless Smartphone and Vehicle Integration to Connect Drivers with Sensors and the Environment in a Holistic Service- Oriented Architecture" which was funded by Ministerio de Economía y Competitividad, Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, Proyectos I+D+I 2014, Spain, under Grant TEC2014-52690-R.
- Analysis of the mobility and information persistence in Vehicular Networks. Aplication in accidents management. Ministerio de Economía y Competitividad, Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, Proyectos I+D+I 2014, Spain, under Grant TEC2014-52690-R, the Generalitat Valenciana, Spain, under Grant AICO/2015/108.
- SETMAN: "Solutions for efficient vehicular traffic management based on networked systems and services", which was funded by the "Ministerio de Ciencia, Innovación y Universidades, Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, Proyectos I+D+I 2018", Spain, under Grant RTI2018-096384-B-I00

8.4 Future Work

In the field of traffic management there are many topics that can be improved. In this thesis, we have investigated the efficient handling of traffic for autonomous vehicles with the possibility of predicting future traffic conditions to optimize the routes used by vehicles within a centralized route management. Thus, we propose some future works.

- The results obtained in this research work can be relevant indicators for researchers and designers of autonomous vehicles that could integrate in their products the capacity of a centralized route management to obtain efficient traffic routes, thus reducing traffic congestion, noise, environmental pollution, delays, and inefficient use of fuel.

- Conduct further experiments to determine the gain achieved by our approach in terms of CO_2 emissions and fuel consumption.
- Link our route server for autonomous vehicles with drones or sensors deployed throughout the streets of a city to obtain real-time information on the behavior of each street, and then apply our algorithms to predict future traffic conditions.

Acronyms

ASCII	American Standard Code for Information Interchange
ABATIS	Automatic Balancing of Traffic through the Integration of Smartphones with vehicles
API	Application Programming Interface
BGP	Border Gateway Protocol
CO_2	Carbon Dioxide
CH	Contraction Hierarchies
CSV	Comma-Separated Values
CDF	Cumulative Distribution Function
DiffServ	Differentiated Services
DLR	German Aerospace Center
GPS	Global Positioning System
HTTP	Hyper Text Transfer Protocol
HCM	Highway Capacity Manual
ID	IDentification
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6

ITS Intelligent Transport Systems

JSON JavaScript Object Notation

LDP Label Distribution Protocol

MAC Medium Access Control

MATSim Multi-Agent Transport Simulation

MANET Mobile Ad hoc NETwork

M2M Machine to Machine

MLD Multilevel Dijkstra

MMTS Multi-agent Microscopic Traffic Simulator

MPLS MultiProtocol Label Switching

NavTeq Navigation Technologies Corporation

NED NEtwork Description

OMNeT++ Objective Modular Network Testbed in C++

OSM OpenStreetMap

O-D Origin-Destination

OSPF Open Shortest Path First

PCA Principal Component Analysis

PSVP-TE Resource Reservation Protocol - Traffic Engineering

PPP Point-to-Point Protocol

QoS Quality of Service

SMARTEST Simulator Modelling Appliyed to Road Transportation European Schema Test

SUMO Simulation of Urban MObility

TAZ Traffic Assigment Zone

UDP User Datagram Protocol

URL Uniform Resource Locator

TCP Transport Control Protocol

TCP/IP	Internet Prototol Stack
TraCI	Traffic Control Interface
VISSIM	Verkehr In Städten - SIMulationsmodell
VISUM	Verkehr Im SIMulationsmodell
XML	eXtendible Markup Language

Bibliography

[1] S. P. Hoogendoorn and P. H. Bovy. “State-of-the-art of vehicular traffic flow modelling”. In: *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 215.4 (2001), pp. 283–303 (cited on p. 7).

[2] J. Harri, F. Filali, and C. Bonnet. “Mobility models for vehicular ad hoc networks: a survey and taxonomy”. In: *IEEE Communications Surveys & Tutorials* 11.4 (2009), pp. 19–41 (cited on p. 9).

[3] A. Jardosh, E. M. Belding-Royer, K. C. Almeroth, and S. Suri. “Towards realistic mobility models for mobile ad hoc networks”. In: *Proceedings of the 9th annual international conference on Mobile computing and networking*. ACM. 2003, pp. 217–229 (cited on p. 9).

[4] V. Naumov, R. Baumann, and T. Gross. “An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces”. In: *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*. ACM. 2006, pp. 108–119 (cited on p. 10).

[5] N. Eagle and A. S. Pentland. “Reality mining: sensing complex social systems”. In: *Personal and ubiquitous computing* 10.4 (2006), pp. 255–268 (cited on p. 11).

[6] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. “SUMO–simulation of urban mobility: an overview”. In: *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. Vol. 42. ThinkMind. 2011, pp. 1–6 (cited on pp. 11, 49).

[7] M. Treiber and A. Kesting. "Car-following models based on driving strategies". In: *Traffic Flow Dynamics*. Springer, 2013, pp. 181–204 (cited on p. 11).

[8] O. Foundatio. *Open Street Map*. URL: https://www.openstreetmap.org/ (visited on 05/10/2019) (cited on p. 12).

[9] H. Technologies. *NAVTEQ*. URL: https://www.here.com/navteq (visited on 05/11/2019) (cited on p. 12).

[10] P. T. V. A. P. Group. *PTV Visum*. URL: http://vision-traffic.ptvgroup.com/es/productos/ptv-visum/ (visited on 05/11/2019) (cited on p. 12).

[11] M. Community. *MATSim*. URL: https://matsim.org (visited on 05/11/2019) (cited on p. 12).

[12] P. T. V. A. P. Group. *PTV VISSIM*. URL: https://www.ptvgroup.com/en/solutions/products/ptv-vissim/ (visited on 05/11/2019) (cited on p. 12).

[13] V. S. GmbH. *OpenDrive*. URL: http://www.opendrive.org (visited on 05/11/2019) (cited on p. 12).

[14] ESRI. *ArcView File - ESRI*. URL: https://www.esri.com (visited on 05/10/2019) (cited on p. 12).

[15] OMNeT++. *OMNeTT ++ Discrete Event Simulator*. URL: https://www.omnetpp.org (visited on 05/11/2019) (cited on p. 13).

[16] OMNeT++. *INET Framework*. URL: https://inet.omnetpp.org (visited on 05/11/2019) (cited on p. 13).

[17] C. T. Calafate, D. Soler, J.-C. Cano, and P. Manzoni. "Traffic management as a service: the traffic flow pattern classification problem". In: *Mathematical Problems in Engineering* 2015 (2015). DOI: 10.1155/2015/716598 (cited on pp. 15, 32, 40–42).

[18] D. G. V. T. of the European Commission. *Simulator Modelling Appliyed to Road Transportation European Schema Test*. URL: http://www.its.leeds.ac.uk/projects/smartest/ (visited on 05/10/2019) (cited on p. 16).

[19] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux. "TraCI: an interface for coupling road traffic and network simulators". In: *Proceedings of the 11th communications and networking simulation symposium*. ACM. 2008, pp. 155–163 (cited on pp. 16, 29, 92).

[20] *Proceedings of the SUMO2017: Towards Simulation for Autonomous Mobility : May 08-10, 2017, Berlin-Adlershof*. Berichte aus dem DLR-Institut für Verkehrssystemtechnik. Deutsches Zentrum für Luft- und Raumfahrt, 2017 (cited on p. 17).

[21] E. Wießner, L. Lücken, R. Hilbrich, Y.-P. Flötteröd, M. Behrisch, L. Bieker-Walz, and J. Erdmann. *SUMO 2018: Simulating Autonomous and Intermodal Transport Systems*. Ed. by E. Wießner, L. Lücken, R. Hilbrich, Y.-P. Flötteröd, J. Erdmann, L. Bieker-Walz, and M. Behrisch. Vol. 2. EPiC Series in Engineering. EasyChair, 2018 (cited on p. 17).

[22] G. A. C. (DLR). *NETCONVERT*. URL: `https://sumo.dlr.de/wiki/NETCONVERT` (visited on 05/13/2019) (cited on p. 17).

[23] G. A. C. (DLR). *DUAROUTER*. URL: `https://sumo.dlr.de/wiki/DUAROUTER` (visited on 05/13/2019) (cited on p. 18).

[24] G. A. C. (DLR). *OD2TRIPS*. URL: `https://sumo.dlr.de/wiki/OD2TRIPS` (visited on 05/13/2019) (cited on p. 19).

[25] G. A. C. (DLR). *ACTIVITYGEN*. URL: `https://sumo.dlr.de/wiki/ACTIVITYGEN` (visited on 05/13/2019) (cited on p. 20).

[26] G. A. C. (DLR). *JTRROUTER*. URL: `https://sumo.dlr.de/wiki/JTRROUTER` (visited on 05/13/2019) (cited on p. 20).

[27] T. V. Nguyen, D. Krajzewicz, M. Fullerton, and E. Nicolay. "DFROUTER-Estimation of vehicle routes from cross-section measurements". In: *Modeling Mobility with Open Data*. Springer, 2015, pp. 3–23 (cited on p. 21).

[28] T. V. Nguyen, D. Krajzewicz, M. Fullerton, and S. T. Mai. "DFROUTER–Route estimate method based on detector data". In: *SUMO2014-Modeling Mobility with Open Data* (2014), p. 127 (cited on pp. 21, 23).

[29] A. Varga. "OMNeT++". In: *Modeling and tools for network simulation*. Springer, 2010, pp. 35–59 (cited on p. 25).

[30] A. Varga. "Discrete event simulation system". In: *Proceedings of the European Simulation Multiconference*. 2001 (cited on p. 26).

[31] D. Delling and R. F. Werneck. "Customizable point-of-interest queries in road networks". In: *IEEE Transactions on Knowledge and Data Engineering* 27.3 (2015), pp. 686–698 (cited on p. 35).

[32] D. Krajzewicz. "Traffic simulation with SUMO–simulation of urban mobility". In: *Fundamentals of traffic simulation*. Springer, 2010, pp. 269–293 (cited on p. 49).

[33] L. Bieker, D. Krajzewicz, A. Morra, C. Michelacci, and F. Cartolano. "Traffic simulation for all: a real world traffic scenario from the city of Bologna". In: *Modeling Mobility with Open Data*. Springer, 2015, pp. 47–60 (cited on p. 49).

[34] H. C. Manual. "HCM2010". In: *Transportation Research Board, National Research Council, Washington, DC* (2010) (cited on p. 58).

[35] H Lieu. "Traffic Flow Theory: A state of the art report-Revised Monograph on Traffic Flow Theory". In: *Turner Fairbank Highway Research Center (TFHRC) at http://www. tfhrc. gov/its/tft/tft. htm* (2002) (cited on p. 76).

[36] S. Menard. *Applied logistic regression analysis*. Vol. 106. Sage, 2002 (cited on p. 79).

[37] A. K. Jain. "Data clustering: 50 years beyond K-means". In: *Pattern recognition letters* 31.8 (2010), pp. 651–666 (cited on p. 81).

[38] I. Jolliffe. *Principal component analysis*. Springer, 2011 (cited on p. 82).

[39] R. Cal y Mayor Reyes Spíndola and J. Cárdenas Grisales. *Ingeniería de tránsito: fundamentos y aplicaciones*. Alfaomega Grupo Editor, 2007 (cited on p. 83).