ISSN: 2227-7994
FORUM A+P
INTERDISCIPLINARY JOURNAL OF ARCHITECTURE AND BUILT ENVIRONMENT

Title: *A Methodology of Traffic Simulation, a Preliminary Work for VANET Technology*

Author: *Tamara Luarasi, Albina Toçilla*

Source: *Forum A+P 27 | Venturing into the Age of AI: Insights and Perspectives*

ISSN: *2227-7994*

DOI: *10.37199/F40002713*

Publisher: *POLIS University Press*

A Methodology of Traffic Simulation, a Preliminary Work for VANET Technology

TAMARA LUARASI
POLIS University
ALBINA TOCILLA
POLIS University

Abstract
Urban regions across the globe continue to grapple with a substantial problem of traffic congestion, which has adverse effects on both the environment and the overall well-being of citizens. Intelligent Transport Systems (ITS) are essential for dealing with and reducing congestion by employing diverse technological solutions. ITS represents a powerful toolbox of technologies and strategies for managing and alleviating traffic congestion in modern cities. Taking into consideration that their real implementation is very expensive, a preliminary simulation work is required.
This paper aims to propose a way for conducting a traffic simulation for an intersection, using real data. It provides a comprehensive description of the necessary steps to take in order to carry out this simulation and conduct subsequent analyses. The simulators used for this purpose are SUMO and VEINS. The findings presented here demonstrate that the configuration of a crucial point in the road network has specific impacts on vehicular traffic and should be taken into account when making decisions about altering or establishing the topology of key points in the road network within a city.

Keywords
Intersection; SUMO; VEINS; performance; geometry

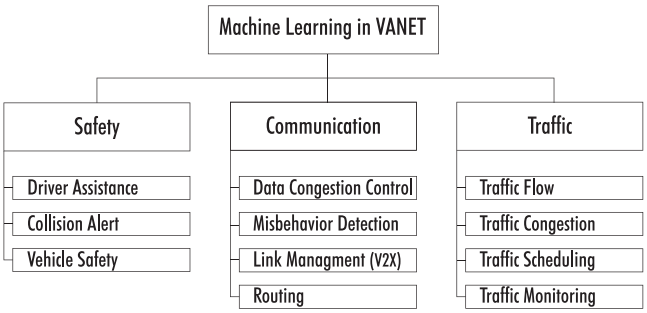

Figure 1: ML use in VANET (Khatri et al., 2021)

Introduction

The vehicular traffic is a significant problem for the urban life. It is a problem for the vehicle, the passenger, and in general, for the life people living in the city. Referring to The Future of Urban Mobility 2.0 study, it is calculated that about 53% of the world's population was living in urban centers in 2014 and an increase of 14% is expected in 2050. This growth of population is associated with the increase of traffic flows in the same time. For this reason, there are a lot of efforts and studies about this problem, that are linked with various aspects of it, like vehicle, accidents and life in general. What is important for the vehicle is the speed of travelling inside the city, the delays caused by the unpredictable traffic, and the possibility of parking when it needs to stop.Vehicular traffic affects both the air quality and life in the city. The rapid advancement of artificial intelligence has brought about a sea change in road traffic management. AI can now predict and control the flow of people, objects, vehicles, and goods at different points of the transportation network with great accuracy. There are a lot of studies that are focused in one of the details of traffic management. Some of them, are focused on the security aspect, to avoid the accidents and the critical cases of the traffic, providing communication networks between vehicles, and between vehicles and RSD units, the so called VANET. Machine learning and fuzzy logic algorithms are also the basis of many VANET clustering algorithms, as we see in the figure (Mchergui et al., 2022). Of course, this phase needs to be preceded by a study of the existing traffic, and its improvement which is linked with urbanistic aspect of the roads too such as the topology of the important points of the road networks like intersections and roundabout. Intelligent Transport Systems (ITS) gain a special attention nowadays. ITS's main aim consist on improving the transportation itself by reducing the traffic problems and providing a sustainable transport system. They include an wide range of communication technologies and smart solutions, which help the traffic flow management and optimization by reducing the congestion. These systems are able to gather and process data related to transportation, to achieve the above aims. One of the key components of an ITS is connectivity, which plays a crucial role in designing an efficient transportation system. These kinds of problems are impossible to be experimented in real life, for different reasons, and on the other side it is quite necessary to forecast the different situations related to the traffic of the city. Then, the simulations of the traffic are essential. There are a lot of software that, by using some parameters based on the real data of the traffic, make possible the generation of a similar traffic with the real one. This article is an effort to represent a methodology to ensure a traffic simulation based on real data for a zone where these data exist. It has a descriptive character of all the steps that we need to follow to achieve this simulation and to make some analyses about it. This methodology also, is illustrated with an experimental work. The results presented here show that the topology of an important point in the road network has certain effects of the vehicular traffic and serve as a practice that people have to take into consideration when they take some decision regarding the topology of some key points

Figure 2: Work Environment interface (Virtual Box)

Figure 3: Veins interface

in the road network, when this topology needs to be changed or defined for the first time in the city.

The Work Environment

It is helpful to create an isolated work environment when we do such a work, to simulate the vehicular traffic. One of the reasons for this, is to avoid the conflicts between the different versions of software that we will download with those that we have already on our computer. For example, we can have differ-

ent versions of Python downloaded in our computer, which may conflict with python version of the SUMO modules.

Virtual Box

A solution for this is the installation of Virtual Box. This software allows us to work inside an operating system Linux quite in isolated way from the other part of computer. VirtualBox is a virtualization software of the type x64. A virtualization software allows one or more operating systems to work inside a computer. It is like having more than one computer. One is physical, the other virtual.(Capterra, 2023) The psychical one it is called the "host;" it can have one of the operating Systems Windows, Linux, macOS, and Solaris, and it supports the other operating systems called "guest" such as Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and Open Solaris, OS/2, and OpenBSD. (Virtualbox) In our work we have used Windows 11 as a host system and Linux as the guest system (the version Debian 11). In this case we installed the version 6.1 of the Virtual Box. Instant Veins, version 5.2, is installed too and imported inside the Virtual Box. In this way the possibility of the execution of Instant Veins directly is provided.

Virtual Machine Instant Veins

When we download Instant Veins, in our case the version 5.2-i1, there are some components that come with it:

- Simulation modules
- Veins 5.2
- INET Framework 4.2.8
- SimuLTE 1.2.0 (plus a backported patch, 23c0936e31)
- Veins_INET included with Veins 5.2
- Software
- OMNeT++ 5.7
- SUMO 1.11.0
- Cookiecutter 1.7.3 for cookie cutter-veins-project
- Operating system
- Debian 11, Linux 5, GNOME 3

The goal here is to study of the road traffic using the features

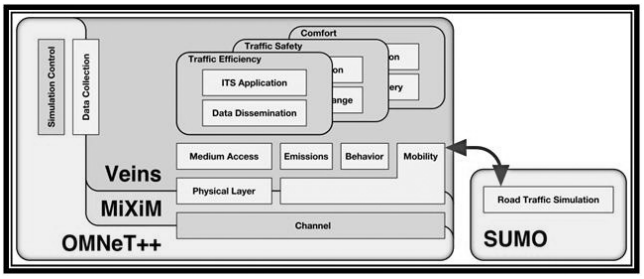

Figure 4: Veins architecture between Omnet++ and SUMO

of SUMO and what OMNET++ provides for this purpose, because it is this component OMNET++ that can fully do the traffic simulation control generated by SUMO and operates closely with this simulator.

Simulation of Urban Mobility (or **SUMO** for short) is a powerful simulator designed to handle a large load network and a specified traffic demand, including a vehicle route and

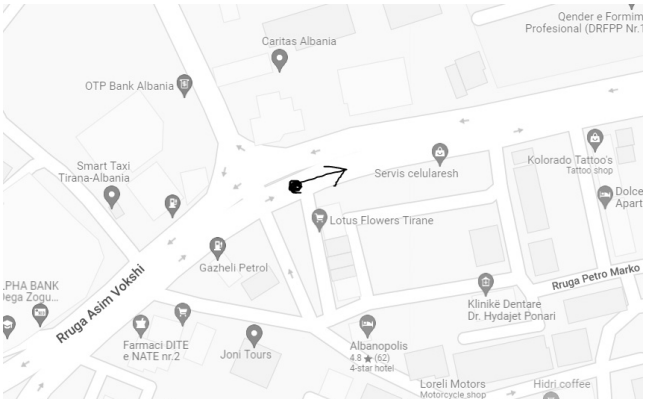

Figure 4: Open Street map view

car following model. It also provides a lot of useful information such as the vehicle speed, model, and position. One of the major features of SUMO is the Traffic Control Interface (or TraCI for short), which is a Python API that treats the SUMO simulation as a server and allows users to gain information from a traffic simulation or modify the simulation. TraCI enables an interface to allow third party systems (or libraries) to integrate with the SUMO traffic simulation.Generally, the traffic patterns or mobility traces created by SUMO can be imported to some of the popular network simulators including Omnet++ to create a realistic vehicle and traffic dynamics. Omnet ++ is the Network simulator that can do the real networking aspects of the simulation and be dealing with the networking components of a typical "network simulation" (such as mac, TCP, routing, etc.,) (Omnet).

Veins is a set of extensions exclusively written for OMNET++. Veins provides a set of protocols to simulate VANET under OMNET++. In addition to that, Veins will work along with SUMO and can use its traffic models (mobility scenarios and patterns) under OMNET++ in a well-integrated fashion.(Omnet)

OMNET++ node for every vehicle is paired with a vehicle moving in SUMO. As a result, simulations of mobility and network can happen simultaneously (bidirectional) with the protocol called Traffic Control Interface or TraCI (Omnet).

The Methodology

[1]*OpenStreetMap (OSM) is a free, open geographic database updated and maintained by a community of volunteers via open collaboration. Contributors collect data from surveys, trace from aerial imagery and also import from other freely licensed geodata sources. OpenStreetMap is freely licensed under the Open Database License and as a result commonly used to make electronic maps, inform turn-by-turn navigation, assist in humanitarian aid and data visualization. OpenStreetMap uses its own topology to store geographical features which can then be exported into other GIS file formats. The OpenStreetMap website itself is an online map, geodata search engine and editor.*

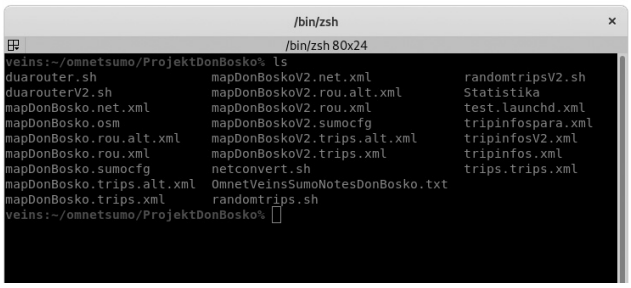

Figure 6: Main files and commands of the project

Using OpenStreetMap , a zone of Tirana is selected, for which is provided some information by sensors regarding the urban traffic for 24 hours, for one week. This information is maintained by the Directory of Transportation in Tirana Municipality.
As a result of the exportation a file is generated, the .osm file with information about the road network of the selected zone.
The second step is the control of this information to avoid the errors that can be caused by different reasons during this process.There are three reasons why converting OSM data directly to a road network for microscopic traffic simulation is problematic: (i) Intersections are not modeled with an explicit data structure in OSM, requiring the conversion process to guess relevant information like intersection geometry, waiting lines, and lane-to-lane connections. (ii) Considering that most of the OSM data is produced by volunteers, human errors are very frequent. Incorrect connections, gaps, misclassifications, and broken ways are common issues found in the road network for a given region, which could result in disconnections and inconsistencies in the traffic simulation. (iii) Variability in how real-world geometry is represented in OSM data is another reason. (Meng et al, 2022)

Generation of the SUMO files for a Simulation

Generation of the Road Network

From the file .osm, in our case mapDonBosko.osm we create the network file that SUMO needs for a simulation.
The command is:
netconvert --osm-files mapDonBosko.osm -o mapDonBosko.net.xml
in fact, we have used for each command a shell file that makes the command shorter
sh netconvert.sh
with content the *netconvert* command.
As the result of this step we get the SUMO network file, in our case mapDonBosko.net.xml,it describes the traffic-related part of a map, the roads and intersections the simulated vehicles run along or across. Nodes, usually named "junctions" in SUMO-context, represent intersections, and "edges" roads or streets. (SUMO)
For example:

<edge id=":4655759636_4" function="internal">
<lane id=":4655759636_4_0" index="0" disallow="tram rail_urban rail rail_electricrail_fast ship"
speed="2.78" length="9.39" shape="444.44,157.28 435.91,161.24"/>

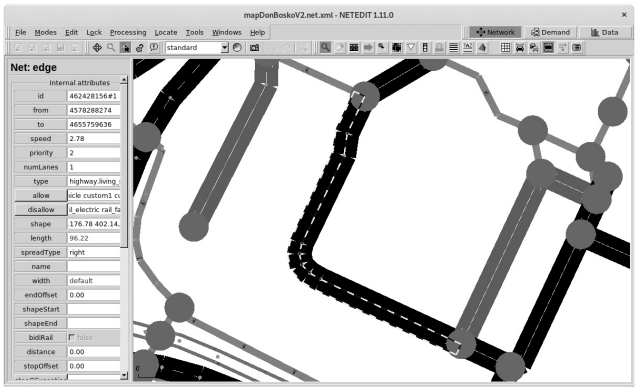

Figure 7: Net.xml file in NETEDIT

</edge>

Represent an ***internal*** edge with id=":4655759636_4", which lies within an intersection and connects and incoming normal edge with an outgoing normal edge. The first part of ID, 4655759636 is the node ID, the edge is located within (see below the bolded id), and 4 is a running number running clockwise around the node (starting in the north). As we see the ID has a ':' as prefix. And the following represent the information related to a normal edge that connects two nodes:

<edge id="-462428156#1" from="4655759636" to="4578288274" priority="2" type="highway.living_street"
shape="439.49,157.82 402.14,175.07 399.20,176.78 397.11,178.96 396.46,181.84 397.10,184.60
408.36,208.05 409.19,213.88 413.85,223.81">
<lane id="-462428156#1_0" index="0" disallow="tram rail_urban rail rail_electricrail_fast ship"speed="2.78"
length="86.44" shape="435.91,161.24 402.87,176.49 400.20,178.05 398.57,179.74 398.10,181.84
398.62,184.07 409.91,207.58 410.74,213.42 414.22,220.83"/>
</edge>

It represents a street (edge) with an id="-462428156#1" that connects two nodes with ID "4655759636" (from), "4578288274" (to), it has the type "highway.living_street", one lane with id="-462428156#1_0" and a length of86.44m, and a shape of the edge is represented by a sequence of points
"439.49,157.82 402.14,175.07 399.20,176.78 397.11,178.96 396.46,181.84 397.10,184.60 408.36,208.05 409.19,213.88 413.85,223.81"
that form the geometry of this edge like in the figure (represented with dashes):
The street name and other features of the edge can be found in the .osm file. Junctions represent the area where different streams cross, including the right-of-way rules that the vehicles have to follow when crossing the intersection.

<junction id="1843955439" type="priority" x="389.02" y="454.54" incLanes="174507820_0 173614254#2_0 -173614254#3_0" in-

tLanes=":1843955439_0_0 :1843955439_1_0 :1843955439_2_0 :1843955439_6_0 :1843955439_4_0 :1843955439_7_0" shape="394.87,459.05 396.23,456.15 394.42,454.81 393.95,453.98 393.79,453.05 393.92,452.01 394.36,450.86 388.60,448.07 383.75,458.24 389.54,460.96 390.89,459.14 391.72,458.68 392.66,458.50 393.72,458.63">

<request index="0" response="000100" foes="100100" cont="0"/>
<request index="1" response="010100" foes="011100" cont="0"/>
<request index="2" response="000000" foes="100011" cont="0"/>
<request index="3" response="010010" foes="010010" cont="1"/>
<request index="4" response="000000" foes="001010" cont="0"/>
<request index="5" response="000101" foes="000101" cont="1"/>
</junction>

Here, we have the id "1843955439" of the junction, the coordinate x and y, the id-s "174507820_0 ", "173614254#2_0", "-173614254#3_0" of the lanes that end at the intersection sorted by direction, clockwise, with direction up = 0, The IDs ":1843955439_0_0", ":1843955439_1_0", ":1843955439_2_0", ":1843955439_6_0", ":1843955439_4_0", ":1843955439_7_0" of the lanes within the intersection, and the shape.

Regarding the ***requests***, the meaning of one of them, for example *<request index="5" response="000101" foes="000101" cont="1"/>* is, the connections 0, 2 (starting from the right) have a higher priority than the connection with index 5 and prohibits un-decelerated passing of the intersection for vehicles at this connection 5. The connection 0, 2 conflict with the connection 5, and the vehicle may pass the first stop line and wait within the intersection until there are no vehicles with higher priority. This is typically the case for left-moving streams from the prioritized direction.
The command netconvert also generates the traffic lights and programs for junctions during the computation of the networks (DLR). The following is this information when it is generated.

<tlLogic id="1842446691" type="static" programID="0" offset="0">
<phase duration="82" state="GG"/>
<phase duration="3" state="yy"/>
<phase duration="5" state="rr"/>
</tlLogic>

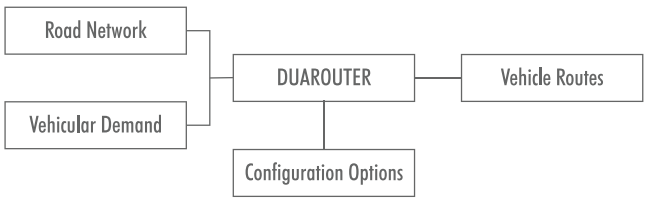

Figure 8: Creation of the routes file

Another information generated by command netconvert are the connections between lanes at junctions (nodes), for example:

<connection from="-1056187988#0" to="551904499#0" fromLane="0" toLane="1" via=":1840291109_3_0" dir="s" state="m"/>

This is a plain connection that describes that outgoing lane "1" of the edge "551904499#0" is reached from lane "0" of the edge "-1056187988#0" via the lane ":1840291109_3_0" across the junction, and the direction of the connection is straight ("s") and the state of the connection is minor ("m"). We also have the following information about roundabout nodes.

<roundabout nodes="1840291020 1840291049 1840291064 1840291157 2523562417 258179027 4539989271 6879850168 6879898516" edges="1056187989 37229363 548351119 549125847 549166140 549166141 549166142 736331977 736331978"/>

Creation of the routes file

Beside the generated network of roads, streets, and nodes, the simulation needs some kind of description about the vehicles. This is called the traffic demand. It is used the following terminology: A ***trip*** is a vehicle movement from one place to another defined by the starting edge (street), the destination edge, and the departure time. A *route* is an expanded trip, that means, that a route definition contains not only the first and the last edge, but all edges the vehicle will pass. There are several ways to generate routes for SUMO.
DUAROUTER is one of them, it generates vehicles' routes based on the demand for individual trips and vehicles' flows, using the shortest route routing algorithms. (Figure 6) (Urquiza-Aguiar et al., 2019) shows the flowchart of this tool, where the road network and the vehicle demand are entered as inputs and as outputs the routes.
In order to create the trip file (Vehicle Demand), SUMO offers the python tool called randomTrips.py. Inside the shell file randomtrips.sh we have the following command:
/home/veins/src/sumo/tools/randomTrips.py -n "mapDonBosko.net.xml" -b 0 -e 3600 -p 6.2 --route-file "mapDonBosko.trips.xml" –validate
*Where mapDonBosko.net.xml is the input file and mapDonBosko.**trips.xml*** is the output file, and the following parameters are used
- n is used to specify the network in this case mapDonBosko.net.xml
- e specifies the end time which is set to 3600 sec
- p represent the arrival rate. The arrival rate is calculated by the formula:(t2-t1)/ n
Where n is the number of vehicles that depart between times t1 and t2, in our case, in 1 hour. For n we have considered an average of the Traffic Vehicles information provided for one week, 24 hour per day. This calculated average is 6.2.
- o specifies where resulting trips are stored

After creating the *mapDonBosko.trips.xml* we need to convert the trips to routes. To do this we will run the shell file duarouter.sh, with the following command inside it:
duarouter -n mapDonBosko.net.xml --route-files mapDonBosko.trips.xml -o mapDonBosko.rou.xml

This will convert our trips into routes. Let's breakdown the options:
- *n* is used to specify the network in this case *mapDonBosko.net.xml*

- ***route-files*** specifies the trip file***mapDonBosko.trips.xml***
- ***o*** specifies where resulting routes are stored, in our case *mapDonBosko.rou.xml*

```
veins:~% cd omnetsumo
veins:~/omnetsumo% cd ProjektDonBosko
veins:~/omnetsumo/ProjektDonBosko% ls
duarouter.sh               mapDonBoskoV2.net.xml           randomtripsV2.sh
duarouterV2.sh             mapDonBoskoV2.rou.alt.xml       Statistika
mapDonBosko.net.xml        mapDonBoskoV2.rou.xml           test.launchd.xml
mapDonBosko.osm            mapDonBoskoV2.sumocfg           tripinfospara.xml
mapDonBosko.rou.alt.xml    mapDonBoskoV2.trips.alt.xml     tripinfosV2.xml
mapDonBosko.rou.xml        mapDonBoskoV2.trips.xml         tripinfos.xml
mapDonBosko.sumocfg        netconvert.sh                   trips.trips.xml
mapDonBosko.trips.alt.xml  OmnetVeinsSumoNotesDonBosko.txt
mapDonBosko.trips.xml      randomtrips.sh
veins:~/omnetsumo/ProjektDonBosko% sumo-gui mapDonBosko.sumocfg
```

Figure 9: The command sumo-gui mapDonBosko. sumocfg

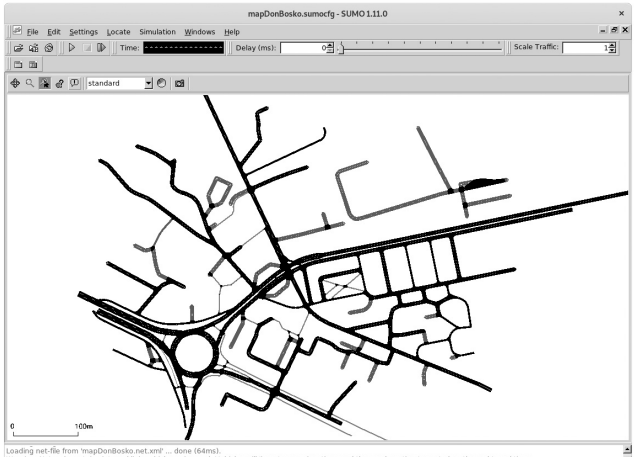

Figure 10: The generated Road Network of the selected zone

The generated information has this character:

<vehicle id="2" depart="12.40">
<route edges="471407810 -471407810 -717156976#4 -717156976#3 -717156976#2 -717156976#1 -717156976#0 173988975#0 173988975#1 173988975#2 173988975#3 173430970#0 173430970#1 459248049 459205964#0 459205964#1 459205964#2 674597415#0 674597415#1 549166141 549166142 549166140 548351119 736331978 210192830#0 210192830#1 551904499#0 712114948#0 712114948#1 462428160#0"/>
</vehicle>

where we see the ID of all the edges that form the road of vehicle with id 2 and start time 12.40s. If we add some other parameters in the command randomTrips.py, we will have additional information about vehicle trip like speed, acceleration, distance between vehicle etc. Finally, the last part is to configure the simulation by specifying the network, routes, and duration in a configuration file, like the following.

<configuration>
<input>
<net-file value="mapDonBosko.net.xml" />
<route-files value="mapDonBosko.rou.xml" />
</input>
<time>
<begin value="0"/>
<end value="86400"/>
</time>
</configuration>

The Traffic Simulation in SUMO

Saving this file as mapDonBosko.sumocfg, we can give the following command directly in console:

sumo-guimapDonBosko. sumocfgand a simulation is generated`

If we add in configuration file the lines

<output>
<write-license value="true"/>
<tripinfo-output value="tripinfos.xml"/>
<tripinfo-output.write-unfinished value="true"/>
</output>

We will get the filetripinfos.xml with the information regarding the trips of one vehicle, one line of it is as follows:

<tripinfo id="3" depart="19.00" departLane="" departPos="0.00" departSpeed="14.29" departDelay="0.40" arrival="31.00" arrivalLane="" arrivalPos="0.20" arrivalSpeed="28.57" duration="12.00" routeLength="119.02" waitingTime="0.00" waitingCount="0" stopTime="0.00" timeLoss="1.41" rerouteNo="0" devices="tripinfo_3" vType="DEFAULT_VEHTYPE" speedFactor="1.00" vaporized=""/>

We will convert this .xml file into one Excel file, thisallow us to create various charts and statistics about speed average of the vehicle in urban arias, or time loss average and other indicators of the traffic.

But also, we can integrate all this work done in SUMO as part of an OMNET++ project.

Running the Simulation from Omnet++

As we mentioned above TraCI enables an interface to allow third party systems (or libraries) to integrate with the SUMO traffic simulation, and one of these is Omnet++, where SUMO plays the role of a server and the client is on Omnet++ side. When we create a project in Omnet++, we are asked to include the Veins project, which is an open-source framework and provides a C++ client library for the TraCI API. A good tutorial for this work is given in. Let's represent very shortly some of the steps of this tutorial to create an OMNET++ project that interact with SUMO.

- A new Omnet++ project from Omnet++ is created

File -> New ->Omnet++ Project, and we accept veins and create and empty project.

- The files generated for SUMO, .net. xml, rou.xml, and. sumocfg files are copied into our project folder

- Another file is needed with the names of these files, let call it "test.launchd.xml"

<launch>
<copy file="mapDonBosko.net.xml" />

---

[2]*We eliminated in our case the lines*
<AnalogueModel type="SimpleObstacleShadowing" thresholding="true">
<obstacles>
<type id="building" db-per-cut="9" db-per-meter="0.4" />
</obstacles>
</AnalogueModel>

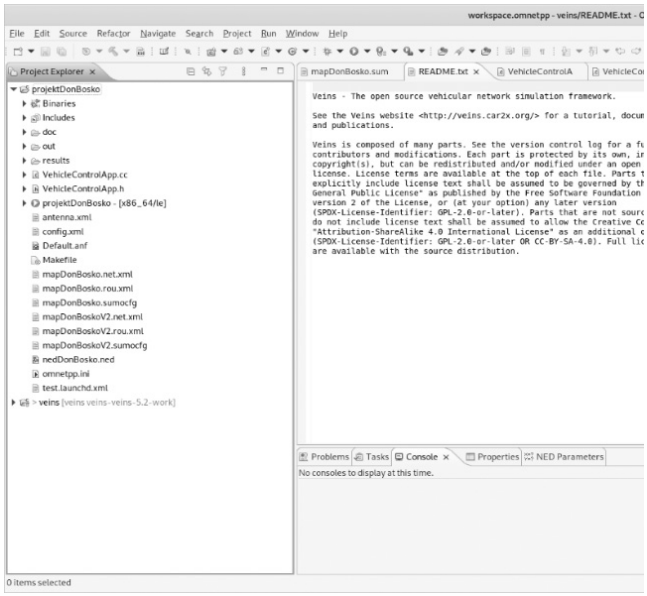

Figure 11: Project explorer interface

<copy file="mapDonBosko.rou.xml" />
<copy file="mapDonBosko.sumocfg" type="config" />
</launch>

As we mentioned, the goal of Omnet++ is the establishment of a vehicle network and this network is tied with the SUMO simulation by some classes of Veins. Veins instantiates one network node per vehicle driving in SUMO. For this reason, we need here the files that we generated for SUMO and some files from Veins folders as is described below.

- *Some files are needed to be copied from Veins example folder [veins_folder]/examples/veins, toour project folder are:*

- *Antenna.xml2:* This file provides a simulation model of antennas that includes parameters for different types of real-world antenna patterns, depending on antenna type, mounting point, and roof topology.

- *Config.xml:* Veins supports some analogue models of signal transmission for the wireless channel, which are defined in this file, and also some other parameters linked with signal transmission.

- The structure of a network simulation model in the NED language (Network Description), and for this e file is created of the type ***.ned***. *File -> New -> Network Description File (NED), empty file*

We copy the content of RSUExampleScenario.ned to our ned file which is:

```
network networkName extends Scenario{
   submodules:
rsu: RSU {
      @display("p=150,140;i=veins/sign/yellowdiamond;is=vs");
       }
}
```

And we change the name of network, putting our name here, nedDonBosko.ned.

- There is also the file ***omnetpp.ini*** where there are included different parameters of different categories that make possible the functionality of the network. To not write everything from scratch we copy the content of the file with the same name from the same folder of veins project and modify some of these parameters as can be shown below. As the result we have the files of our project as they are shown in the Fig 11.

- ***Modification of the file omnetpp.ini***

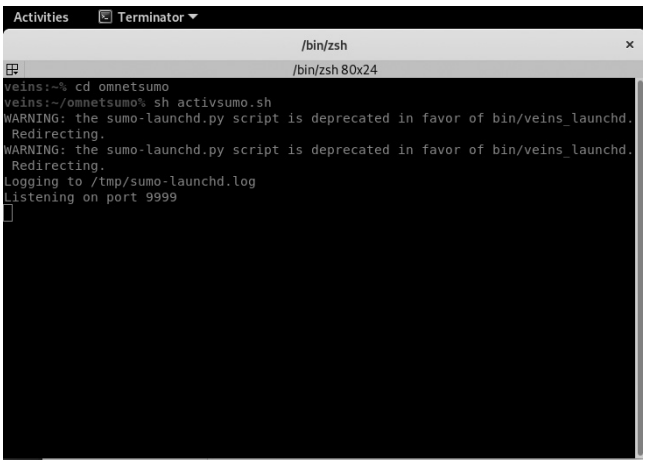

Figure 12: The way how to activate sumo from terminal

To run the simulation using the file omnetpp.ini. The file tells the simulation program which network to simulate, allows you to assig values to different parameters declared in .ned files (your file or inherited ones). Many other parameters are included here

Figure 13: Execution process

which can be grouped by some categories some of them are:

- General parameters
.. . .
network = nedDonBosko
. . . .
- Simulation parameters
- Obstacle parameter (we have eliminated this part)
- TraCIScenarioManager parameters
. . .
manager.launchConfig = xmldoc("test.launchd.xml")
. . .
- RSU Settings
- 11p specific parameters
- App Layer parameters
- Mobility parameters

We are leaving all the other with default values, because our goal is only what statistics Omnet++ offers to simulated traffic of SUMO.

- Reference to Veins libraries

There will be lots of errors because the Omnet++ simulator is a network simulator. By default, it is not aware of many objects like RSUs, Cars, etc., which are implemented in Veins

We need to reference to Veins libraries that developers have made.

Right-click the project in the project explorer (in our case projectDonBosko) Properties -> Project References -> Click Veins -> “Apply and Close”

- Run the project

Then we are ready to run the project from Omnet++. But before we have to launch sumo in parallel, so that is can wait for incoming connections on the port specified in the behavior of our application (generally 9999). Therefore, we need to start the TraCI server first by the command:

python /home/veins/src/veins/sump-launch.py -vv -c sumo-gui

and we will see the screen

Now right click on omnet.ini, run->Omnet++ simulation and we will see the view of Figure 13. We answer OK and push the button Run or Run Fast and the GUI of SUMO of Fig 10 will be shown and push the button Run again and the traffic is generated.

The outputs

And we can see the results afterwards. It is a folder in project that will be generated with the name results, where there are generated multiple files. We double click *.sca and *.vec file and an output file (with extension.anf) will be generated We go to “scalar” or “vectors” tab, and select the data we want to display, right click, plot, and we will have the corresponding graph. All these results are additional results that we can get for the traffic and help us create a better view for the traffic. We can also export this information and get a .cvs file to find different statistics from it.

From OMNET++project we can also use the same outputs that we could get directly from SUMO. For example, adding in the mapDonBosko.sumocfg of the OMNET++ project the lines

<output>
<write-license value="true"/>
<tripinfo-output value="/home/. . ./tripinfos.xml"/>
<tripinfo-output.write-unfinished value="true"/>
</output>

We can get the file tripinfos.xml with trip information in the folder /home/. . ./. We must note here that if you leave the default folder for this file, it is not possible to get it, so use another folder for this goal.

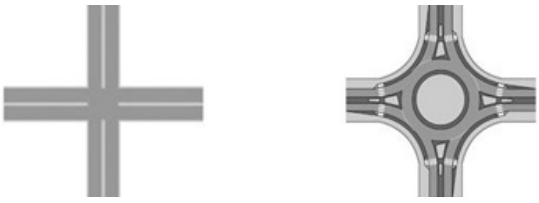

Figure 14: Two geometric types of intersections

Experimental Work

The experimental work here is related to the topology aspect of the road network, how this topology influences on the vehicle traffic. Recent decades have been faced with a significant amount of research focused on the intricate geometric patterns found in many cities, as these patterns provide a means of comprehending the development of cities through a range of methodologies. The cities' gradual development and structural organization have had an impact on certain areas at the macro scales. As a result, a typical city's entire structure has evolved gradually, from small to large scale, and with a large-scale connectivity among different parts. Starting from the geometric complexity and urban morphology of the city, it is necessary to improve the urban systems understanding. Recent decades have seen a significant amount of research focused on the intricate geometric patterns found in many cities, as these patterns provide a means of comprehending the development of cities through a range of methodologies. The core of building an urban terrain model is its morphology. Urban roads have unique geometric and semantic properties because they form the framework of the urban terrain. The road is a strip-shaped feature that is connected and interconnected. It is flat in the horizontal direction and gently undulating in the vertical direction. (Smith, Jonhnson, 2020) Intersections are essential to the efficient flow of traffic on the extensive network of roads that crisscross our cities and towns. These are the locations where two or more roads converge, enabling safe crossings for both cars and pedestrians. Road intersections come in different varieties, each with unique features and purposes. We will examine the characteristics and

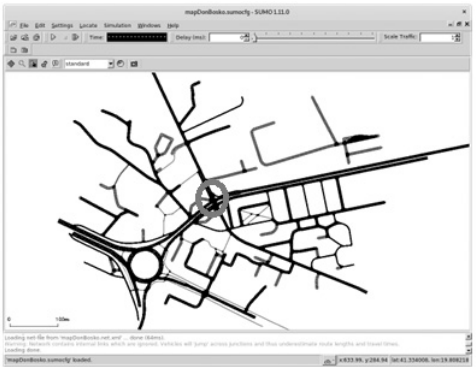

Figure 15: The intersection where the topology is changed

Attribute	Value	Explanation
Length	2.5	Physical length of vehicles
Accel	2.6	The acceleration ability of vehicles of this type (in m/s^2)
Decel	4.5	The deceleration ability of vehicles of this type (in m/s^2)
Sigma	0.5	0≤sigma≤1; driver imperfection; The behavior of sigma is carFollowModel dependent
Tau	1	This parameter is intended to model a drivers desired minimum time headway (in seconds).
minGap	2.5	Empty space after leader [m]
maxSpeed	55.55	The vehicle's (technical) maximum velocity (in m/s)
departSpeed	Max	Determines the speed of the vehicle at insertion, where maxSpeed = MIN (speedLimit * speedFactor, vType_desiredMaxSpeed * speedFactor, vType_maxSpeed):
departLane	Best	Determines on which lane the vehicle is tried to be inserted;
departPos	random	Determines the position on the chosen departure lane at which the vehicle is tried to be inserted;
b	0	Begin time
p	0.1-1	The arrival rate (also know as departure rate or insertion rate) is controlled by option –period (default 1). By using value below 1, multiple arrivals per second can be achieved.
e	3600	Simulation time/end time

Figure 16: Simulation parameters used

functions of some of the most prevalent kinds of road intersections in this section.Critical locations for interactions between cars and pedestrians are road intersections, where traffic patterns must be carefully controlled. Traffic engineers and city planners can create effective and secure transportation systems by having a thorough understanding of the various kinds of road intersections.[19]Two types of intersections (four legs and roundabout), Figure 14 are considered as case studies for the purpose of this article.

A four-way intersection is composed of two perpendicular roads. It is also known as a crossroad. They create a shape akin to a square or cross, enabling traffic to gather in the middle from all four directions. At four-way intersections, traffic signals or stop signs are frequently used to control the flow of cars. Circular intersections called roundabouts are made to improve traffic flow and lower the chance of accidents. They have a central island with constant traffic moving around it in an anticlockwise direction. Roundabouts rely on yield-at-entry laws and do not require stop signs or traffic signals. They enhance safety and facilitate efficient traffic flow. NETEDIT, a SUMO component, provides us the possibility to change the topology of an intersection. As it is shown in Fig 15 the shape of an intersection is changed by NETEDIT. The used simulator SUMO and the simulation parameters are the same in both cases (as in the following table 1). The results are analyzed in terms of arrival speed, depart speed and time loss. The comparison of these parameters in both intersections are demonstrated in the charts below and the difference between them is a clear one. For the effect of experimental work, here there are used some more parameters that provide an improvement of the traffic features, and especially the number of the cars in traffic. Some values of the arrival rate are used in the interval 0.1-1 that generate different numbers of vehicles in traffic. The simulation is done using the same parameters (demonstrated in the following figure). The majority of them are used in their "default" values. The parameter "p" is changed from 0.1 to 1, in order to change the urban traffic flux in the intersection, while the simulation time is 1 hour (equal to 3600 seconds).

p	arrivalSpeed (m/s)	departSpeed (m/s)	duration	waitingTime	timeLoss	id max
0.1	0.79	8.5	1592.6	291.58	890	35942
0.3	2.86	8.49	979.25	223.99	571.23	11995
0.5	5.63	8.69	391	84.39	168.81	7193
0.7	9.2	9.03	69.4	0.46	0.95	5141
1	9.25	0.01	68.65	0.26	0.55	3599

p	arrivalSpeed (m/s)	departSpeed (m/s)	duration	waitingTime	timeLoss	id max
0.1	1.58	6.48	1546.6	232.5	819.2	35942
0.3	3.81	5.85	622.2	112.99	302.78	11995
0.5	6.69	6.22	54.54	0.84	2.21	7193
0.7	6.77	6.33	52.62	0.36	0.96	5141
1	6.9	6.3	51.88	0.19	0.51	3599

Table 1: The results for roundabout and four-legs intersections

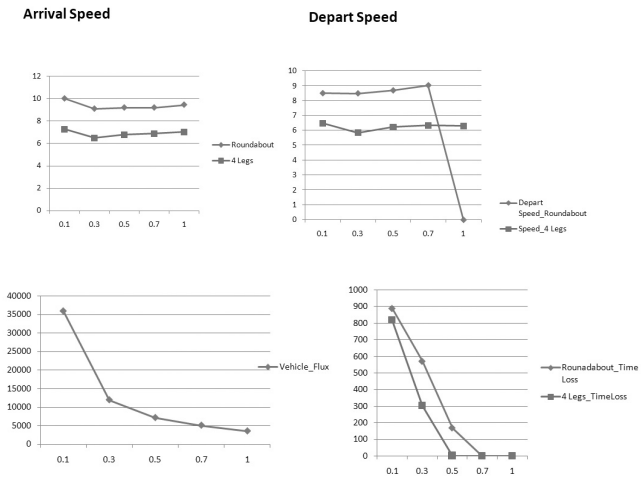

Figure 17: The results in terms of speed and time loss

The model used for the simulations is Krauss because it was specifically chosen to guarantee a speed that consistently maintains the minimum gap (minGap), whereas other models may not prioritize this requirement. The car following model will automatically adjust its driving speed to minimize the need for braking until the maximum limit set by the deceleration function is reached. In the case of the default, there are strict constraints on the Krauss model, whereas for alternative models such as IDM, the deceleration bounds are less strict. There are several ways to view different aspects of traffic, including traffic congestion. It can be characterized as an occurrence of longer vehicle lines, slower driving and longer travel times than typical in terms of both speed and duration. The table below summarizes the simulation results in terms of arrivalSpeed, departSpeed, duration, waitingTime and timeLoss. The column of id_max shows the total number of vehicles in the intersection. The data processing is realized by using R Studio. The difference between two types of intersections is supported by the provided data results, where the four-legs one is dominant as the timeLoss is less compared to the roundabout and also waiting time or speeds follow the same logic. These data are shown in the following graphs, focusing only on arrival speed, depart speed and time loss.

Intersections are frequently referred to as bottlenecks in traffic systems because of a variety of factors that contribute to congestion and a reduction in overall vehicle flow. From this point of view, it is important to address the traffic issues in different morphological types of intersections. The results presented here show that the topology of an important point in the road network has certain effects of the vehicular traffic. It seems that, the four-legs intersection seems to respond better to the same traffic flux generated in sumo, compared to the roundabout one.

Conclusions

The urban vehicular road traffic is an important problem for the city and for this reason, it is important to have the possibility to trace and forecast it. The simulation of the traffic makes this possible. During rush hour, busy intersections where multiple roads intersect can become bottlenecks. Congestion is a consequence of signal timing, geometric design and vehicle volume.

This paper describes a methodology step by step of how to achieve this simulation in a zone of the city, and how to get some results that will serve us for different goals. The methodology suggests a virtual work environment, and inside it the work with SUMO& OMNET++ &Veins. In this paper we are focused on the work with SUMO and SUMO & Omnet+++Veins, but in the second case again, we are focused only on the results related to traffic. We keep everything default regarding Omnet++ and Veins. The experimental work is related to the effects of the topology of an important point of the zone in relation to the traffic of this zone. Therefore, the actors of the urban planning, when they take some decision regarding the topology of some key points of the city, have to have as a practice consideration of

Reference List

Capterra. (2023.). Virtualization Software: A Comprehensive Guide. Retrieved from https://www.capterra.com/resources/virtualization-software/

VirtualBox. *VirtualBox*. Retrieved February 12, 2024, Retrieved from www.virtualbox.org/

https://www.projectguideline.com/vehicular-adhoc-network-vanet-simulation-with-veins-omnet-sumo/ Retrieved February 12, 2024

OMNeT++ Community. (February 10, 2024.). SUMO VANET Tutorial. OMNeT++ Community. https://omnet-manual.com/sumo-vanet/

Meng, Z., Du, X., Sottovia, P., Foroni, D., Axenie, C., Wieder, A., Eckhoff, D., Bortoli, S., Knoll, A., & Sommer, C. (2022). Topology-Preserving Simplification of OpenStreetMap Network Data for Largescale Simulation in SUMO. SUMO Conference Proceedings, 3, 181–197. https://doi.org/10.52825/scp.v3i.111

"SUMO Road Networks Documentation" Retrieved from https://sumo.dlr.de/docs/Networks/SUMO_Road_Networks.html, Accessed February 2, 2024

DLR Institute of Transportation Systems. "Traffic Lights - Simulation of Urban MObility (SUMO)." SUMO - Simulation of Urban Mobility, https://sumo.dlr.de/docs/Simulation/Traffic_Lights.html

Introduction to demand modelling in SUMO. https://sumo.dlr.de/docs/Demand/Introduction_to_demand_modelling_in_SUMO.html

Urquiza-Aguiar, L. F., Coloma Gómez, W., Barbecho Bautista, P., & Calderón, X. (2019, November). Comparison of traffic demand generation tools in SUMO: Case study: Access highways to quito. In *Proceedings of the 16th ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks* (pp. 15-22).

duarouter - SUMO Documentation. (2022). Retrieved from https://sumo.dlr.de/docs/DUAROUTER.html Accessed January 10, 2024

Custom Veins Example Prof. Sangyoung Park Module "Vehicle-2-X: Communication and Control", Retrieved from http://cse.iitkgp.ac.in/~soumya/micro/t2-3.pdf

www.clickworker.com/customer-blog/artificial-intelligence-road-traffic/

Mchergui, A., Moulahi, T., & Zeadally, S. (2022). Survey on artificial intelligence (AI) techniques for vehicular ad-hoc networks (VANETs). *Vehicular Communications*, 34, 100403.

https://veins.car2x.org/documentation/modules/

Khatri, S., Vachhani, H., Shah, S., Bhatia, J., Chaturvedi, M., Tanwar, S., & Kumar, N. (2021). Machine learning models and techniques for VANET based traffic management: Implementation issues and challenges. *Peer-to-Peer Networking and Applications*, 14, 1778-1805.

[19] Structural Guide. (2020). Types of Road Intersections. 13, 1369–1382. Retrieved from https://www.structuralguide.com/types-of-road-intersections.