A Survey on Network Simulators for Vehicular Ad-hoc Networks (VANETS)

Israa A. Aljabry
Department of Computer Engineering
University of Basrah, Basrah, Iraq

Ghaida A. Al-Suhail
Department of Computer Engineering
University of Basrah, Basrah, Iraq

ABSTRACT

In recent years, the number of vehicles has been increasing enormously, this growth has led to an increase in the possibility of accidents. To overcome this situation and to make the driving experience enjoyable, and more comfortable, a special technology based on the Mobile Ad-hoc wireless network infrastructure (MANET) has appeared on the surface. Such technology is called Vehicular Ad-hoc Networks (VANETs). It allows vehicles to communicate with each other or communicate via infrastructure to exchange different kinds of information and warning messages. Scientists and programmers have developed VANET simulation programming to enable the examination and evaluation of various media access, routing, and crisis alerting protocols. In this paper, a short survey is pursued on some common simulators that can be applicable in VANETs. Because such platforms can save a lot of money and time, as well as make the simulation process more realistic; OMNeT++ and SUMO are both used together to investigate the VANET environment using a road traffic simulation. The simulation results obtained can draw a good and helpful guideline in developing VANET performance through its routing protocols.

Keywords

VANET, Simulators, Routing Protocols

1. INTRODUCTION

Since the 1970s, the work on the Ad-hoc Networks has launched; and is initially known as packet radio networks. It mainly relies on the concept of creating a temporary wireless network between moving nodes. Nowadays, such networks are developed to support vehicle driver via the growth of modern applications and business models. Specifically, the development of the Fifth Generation (5G) networks is becoming more readily available. With the advancement of next-generation intelligent vehicle networks and applications, Vehicular Ad hoc Networks (VANETs) and Software Defined Networking (SDN) constitute the primary enablers of 5G technology [1-4]. In particular, the 5G based ad-hoc vehicular network (5G-VANET) plays a promising role in promoting numerous implementations of the Intelligent Transport System (ITS) [1-5]. There are three types of 5G-VANET networking modes: cellular, reuse, and dedicated mode. Additionally, an important development of Vehicular Ad Hoc Networks (VANETs) has been brought on by recent developments in cellular technologies as well as the Internet of Things (IoT). In the new Intelligent Transport System (ITS), the main elements are VANET and IoT. With the support of IoT, a network of vehicles that can communicate with other vehicles and RSUs can be built up [1-4]. However, with the sharp increasing road vehicles in recent years, driving has become very challenging and dangerous. Notice that routes are filled, safety distance and fair speed are not quite possible respected; besides the drivers do not always receive adequate attention. To resolve these challenges and troubles, a good communication network must be established between the vehicles through the technology of VANET. Today, several auto institutions are transporting vehicles with in-car sensors, figurines, and remote specialized gadgets installed, and navigation systems (e.g., GPS) to prepare for the use of large-scale vehicle networks. Using different sensors, cameras, and communication skills, vehicles can gather and interpret information to help the driver to make a judgment, in particular in driver support systems.

In general, VANET communication can be classified into two categories: (*i*) Vehicle-to-Vehicle (V2V) Communication, and (*ii*) Vehicle to Infrastructure (V2I) Communication. In V2V, the communication occurs between different vehicles with On-Board Unit (OBU) devices. Meanwhile, V2I is communication between the vehicles with OBU and Road-Side Unit (RSU). This communication is carried out using various electromagnetic waves such as infrared, microwaves, and radio waves. In this context, many recent publications over two decades were proposed for studying, developing, and improving routing protocols in VANETs using various types of network simulators such as NS-2, NS-3, and OMNeT++; and they have dealt with multi metrics such as distance, density, link stability, speed, and environment changes due to vehicle mobility [9].

This paper is organized as follows: Section 2 includes context for VANET vs. MANET and a short review on routing protocols, Section 3 explains briefly the features of some simulators that can be handled for VANETs focusing on vehicular mobility generators. Section 4 introduces the main network simulators used in performance analysis in different types of wireless networks involving VANETs. In Section 5, and Section 6, the proposed VANET model has been illustrated using Veins and OMNeT++ to provide the simulation methodology of such network via the obtained results. Finally, the conclusion is drawn in Section 7.

2. BACKGROUND

2.1 VANET vs. MANET

MANET (Mobile Ad-hoc Network) and VANET (Vehicular Adhoc Network) are attracted by their usability. Vehicle Ad hoc Networks (VANETs) is a subsection of MANETs [6] (Mobile Ad Hoc Networks). The node's movement is predicted as the vehicle travels on the road in a random path, Unlike in MANET, i.e. the patterns of motion are limited by the topology of the road. VANET is very much a promising field of research interesting to the Scientific Community, auto manufacturers, and mobile internet. Table 1 demonstrates a comparison between VANET and MANET.

2.2 ROUTING PROTOCOL IN VANET

Routing protocols are classified depending on (*a*) the type of network architecture and (*b*) the mode of operation.

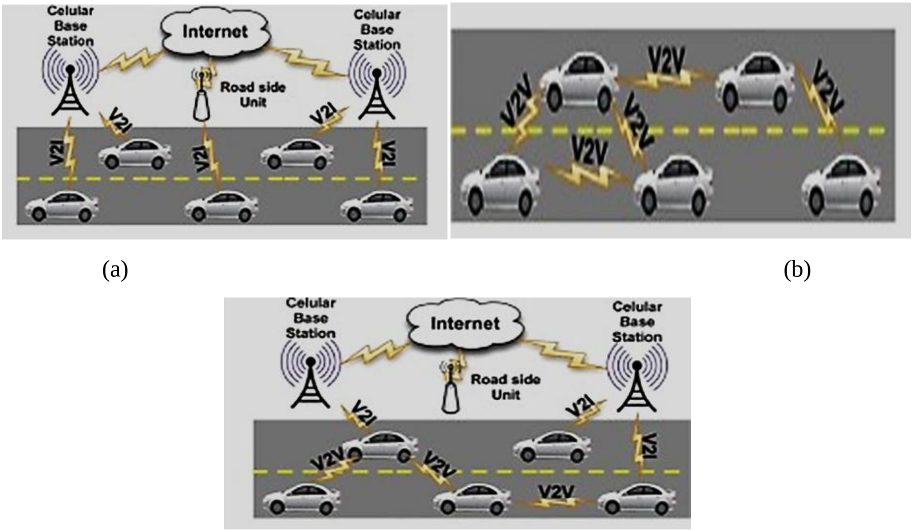
Table 1. Comparison between VANET and MANET

	VANET	MANET
Bandwidth	1000 Kps.	100 Kps.
Transmission Range	500 m	100 m
Cost of production	Expensive	Cheap compared to VANET
Nodes Lifetime	Depends on vehicle lifetime	Depends on power sources
Acquisition of Position	Using GPS, Radar	Using Ultrasonic.

A. Type of Architecture

The type of architecture includes the network topology that should be taken into consideration to perform the routing protocol. Therefore, such architecture may be divided into ad hoc, infrastructured, or hybrid, as shown in Figure 1. The infrastructure mode is shown in Figure 1(a), which is often known as Vehicle-to-Infrastructure (V2I). In this mode, VANET simply uses access points in such a fashion whereas the vehicles can access the Internet and the traffic; and the routing data information can be obtained. Communication is made only between the vehicles and the set Access points. This form of architecture provides connectivity to the mobile nodes; yet, it can be unfeasible if take into account the expenses of infrastructure and edification. In the ad hoc mode of Figure 1(b), also known as the Vehicle-to-Vehicle (V2V) mode. Mobile networks can share information directly with each other with no need for a central access point. The nodes must be able to communicate with each other, working as routers and transmitting messages from the origin node to the destination.

For instance, such architecture could be used to avoid accidents at intersections without traffic signals, alerting traffic jamming, accident detection, and detecting traffic slowdown. Hybrid architecture in Figure 1(c) combining the infrastructure and ad hoc architectures has also been used in VANETs. This type of architecture is one of many applications that can be used in such a way to allow the vehicles to exchange information, and it may also be connected through the Internet via fixed access points [7-11].



(a)

(b)

(c)

Fig 1: Network architectures for VANETs. (a) Infra-structured. (b) Ad hoc (c) Hybrid [12]

B. Mode of Operation. The mode of operation comprehends the type of routing. It can be divided into topology and geographic [13]. Figure 2 depicts an example of the classification of VANET routing protocols. Topology-based routing protocols work in either the unicast mode or multicast mode; and they can be classified into two main groups: (*i*) proactive and (*ii*) reactive. Proactive protocols can maintain the information about the network topology, and they can continuously update the network topology information in routing tables in the nodes whatever the usage of the routes stored. For example, Destination-Sequenced Distance Vector (DSDV) [13] is one example of proactive protocols. Reactive protocols do not sustain routing information updated, and they locate routes only when a source node needs to transmit data packets to a destination node. The Ad hoc On-demand Distance Vector (AODV) routing is an example of reactive protocols [13].

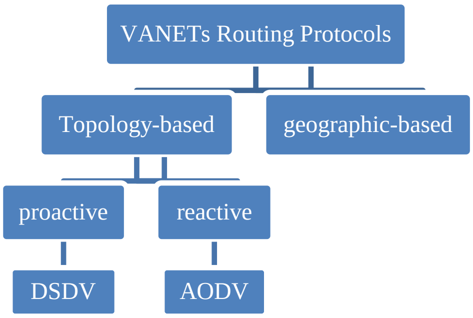


Fig 2: Routing protocols in VANETs

3. MOBILITY SIMULATOR IN VANET

In this section, multiple VANETs simulators that are available to the public use within the research community are reviewed. In this study, the VANET performance is studied via the vehicular (*a*) mobility generator, (*b*) network simulator, and (*c*) VANET simulator. Basically, the mobility generators are required to enhance the realism degree in simulations with VANET. They build realism traces of vehicular mobility to be used as an input for the simulator network. The mobility generator inputs include a model for the road, scenario parameters (i.e., maximum vehicle speed, check-in, and departure dates, etc.). The trace output details where each vehicle is located for the entire simulation time at each instant and their profiles on mobility, for example, SUMO, MOVE, CityMob, STRAW, Netstream (NETwork Simulator for TRaffic Efficiency And Mobility), FreeSim, and VanetMobiSim [11]. To be more realistic, the mobility model should take account of the following factors [14].

1. Accurate and realistic topological maps: The model should consider actual maps containing various types of streets with differing densities of vehicles and different speeds of network nodes.

2. Vehicles acceleration and deceleration: Network vehicles do not immediately stop and drive, so acceleration and deceleration can play an important role.

3. Obstacles: mobility and wireless communication obstacles.

4. Attraction points: At these attraction points, some points are the final destination of several drivers that may cause bottlenecks. It can be considered that the drivers drive from repulsion points to those of attraction.

5. Weather and traffic conditions: Weather conditions, and traffic conditions such as rush hours, weekends, and going to and from schools, play an important role.

6. Driving patterns: In addition to static obstacles, drivers deal with complex obstacles such as adjacent vehicles and pedestrians, so the mobility model should be able to manage all the surrounding circumstances affecting the network.

3.1 SUMO

Simulation of Urban Mobility (SUMO) is a free, downloadable, microscopic, and continuous traffic simulation package designed to manage large road networks [15]. It enables intermodal traffic networks to be modeled including road vehicles, public transport, and pedestrians. SUMO includes a wealth of supportive tools to handle tasks such as finding routes, visualizing, importing networks, and emission calculating. SUMO supports the import formats OpenStreetMap, NavTeq MATsim, VISSIM, VISUM, and OpenDRIVE. SUMO is implemented in C++ and Python and it uses only portable libraries. Also, SUMO can be used on a variety of platforms such as (Windows, Linux, or macOS) integrated with the network simulator like (OMNeT++, NS-2, or NS-3).

3.2 MOVE

Mobility Model Generator for Vehicle Networks (MOVE) is a mobility generator created on top of the SUMO simulator [16] to create practical traces of mobility that are suitable for input into network simulators such as NS-2 and QualNet. MOVE is implemented in Java, with two main components [17]:

1. Map editor: This is used to construct a road topology. Maps can be manually or automatically generated or imported directly from Google Earth.

2. Vehicle movement editor: Help users identify the routes of the trips that the network vehicles can take. It determines not only the routes but also other properties of the route such as the number of vehicles within the route, the vehicle's time of departure, the vehicle's destination, the duration of the journey, and other properties.

3.3 CityMob

CityMob Mobility Generator is an open-source simulator that enables users to create urban mobility scenarios including simulating incidents and using flood-based broadcast events. It produces traces of mobility of vehicles compatible with the NS-2 network simulator. Also, it is implemented using C programming under the GNU license. CityMob proposes three separate models of mobility: (*i*) Simple Model (SM), (*ii*) Manhattan Model (MM), and (*iii*) Downtown Model (DM) [18].

3.4 STRAW

STreet RAndom Waypoint offers precise simulation results using mobility in a vehicle in real US cities model and real vehicle traffic. Existing STRAW implementation is written for JiST / SWANS discreet event Simulator, and traces of its mobility cannot be reliable. Other network simulators included, such as NS-2. STRAW is a member of the C3 (Car-to-Car cooperation) project. In this scenario, opting for an additional practical mobility model with an appropriate level of detail for vehicle networks becomes necessary for the accurate network simulation. The disadvantage of this approach is that the model will identify node movements to only the streets predetermined by map knowledge as in the case of countries in North America. This also guarantees restricted mobility for individual vehicle congestion and simplified traffic management mechanisms.

3.5 FreeSim

FreeSim is a macroscopic completely customizable and microscopic free-flow simulator, allowing multiple freeway networks for quick visualization. and loaded as a graph data structure into the simulator with edge weights defined by current speeds. This platform can build and execute traffic and graph algorithms. For the whole network, or single vehicle or nodes, and traffic data that the simulator uses will be created by the user or converted from real-time data collected by a transport organization. Vehicles found in FreeSim can communicate with the monitoring system the freeway traffic. This makes FreeSim suitable for simulation of the Intelligent Transportation System (ITS). FreeSim is licensed under the General Public License (GPL) and the source code is freely available for download.

Table.2 A Features Comparison of the Various Types of Mobility Generators

	SUMO	MOVE	CityMob	STRAW	FreeSim
Open-source	Yes	Yes	Yes	Yes	Yes
GUI	Yes	Yes	Yes	Yes	Yes
Real maps	Yes	Yes	No	Yes	Yes
NS-2 support	Yes	Yes	Yes	No	No
QualNet support	No	Yes	No	No	No
Macroscopic	No	No	No	No	Yes
Microscopic	Yes	Yes	Yes	Yes	Yes
Continuous development	Yes	No	Yes	No	-
Available examples	Yes	Yes	No	-	Yes
Ease of setup	Moderate	Easy	Easy	Moderate	Easy
Ease of use	Hard	Moderate	Easy	Moderate	Easy

4. NETWORK SIMULATORS

There are various types of simulators: (*i*) commercial and (*ii*) open-source simulators. The source code of the commercial software is not free and is licensed by the company or community owner. On contrary, open-source simulators are free to be used by researchers and consumers without charge for the use of the simulator software [9]. Commercial simulators have an advantage over open-source simulators since they have some professionals, who work in a dedicated way to correct any bugs in this simulation and provide users with continuous support, including the simulator's available documentation. Open source simulators have their benefits as they are adaptive to very quickly reflect new and current technology and topologies compared to the licensed simulators but they are low in documentation and support. Other types of network simulators are simple and complex simulators, in which everything depends on the existence and ease of use of Graphical User Interfaces (GUI) and the command-line interface (CLI). In vehicular ad hoc networks (VANET), network simulators are used to implement and simulate the proposed network with the traces generated by the vehicular mobility generator to achieve more practical results that help to evaluate the network's behavior by applying any protocol or algorithm. Many network simulators are used for this task of simulating the ad-hoc networks of the vehicles. In this section, some of these network simulators are discussed as follows.

4.1 NS-2

NS-2 is a discrete simulation of events developed by a study group for the VINT project at the University of California at Berkeley [18]. The Simulator has been expanded at the Carnegie Mellon Monarch Research Group University to include: (a) mobility of nodes, (b) A functional physical layer with a radio propagation device Model, (c) interacts with the radio network, (d) IEEE 802.11 MAC (Medium Access Control) protocol that uses the Distributed Coordinating Feature (DCF).   The NS-2 distribution code, however, did have some significant defects in both the overall design and modeling 802.11 MAC and PHY. Network simulator version 2 (NS-2) uses two different languages, C++ and OTCL (Object Oriented Tool Command Line), where C++ specifies the internal structure and mechanism of the simulation artifacts.   OTCL describes the external simulation environment to configure the objects. In the installation of the software, NS-2 has some limitations in running on windows, software programs to build UNIX such as environment, and this software program is known as Cygwin, which has a complicated installation process due to the large size of Cygwin packages.

4.2 NS-3

Network Simulator NS-3 is a discrete-event network simulator for Internet systems. It is targeted mainly for academic and educational purposes. NS-3 is free software licensed under the GNU GPLv2 license and freely available for study, development, and use. However, it is not an extension of the Network Simulator version 2 (NS-2) but a simulator designed from scratch that is still under development away from the NS-2 roadmap. NS-3 simulator is written in C++ language and python, networks can be implemented in pure C++ language and some parts of simulation can optionally be implemented in python. The OTCL (Object Oriented Tool Command Line) APIs used in NS-2 simulators are not used by NS-3 simulators [17-22].

4.3 OMNeT++

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators [23]. OMNeT++ is a discrete event simulation environment developed by Andras Varga at Budapest University of Technology, which models primary communication networks. It is also used in other recent applications such as complex IT model simulation, multi-processor modeling, and queuing systems due to its generic and scalable architecture [9]. This simulator fills the gap between the NS-2 open-source simulator and the OPNET license-based simulator by providing free software with different features, providing component-based, modular, hierarchical, and extensible architecture. The modules and components of OMNeT++ are designed using the C++ language, and new versions of the software program the modules using the C++ class library consisting of the simulation kernel.

On the other hand, Network Definition (NED) is also a high-level language used in larger modules and models to collect single components. A graphical network editor (GNED) for the simulation environment includes a NED compiler, command line (Cmdenv), graphical (Tkenv), and graphical performance analysis software (Plove), and a documentation tool. In different fields, researchers have written numerous simulation models and model systems for OMNeT++ over the years. Most of these model frameworks are open source, built as independent projects, and follow their release cycles. The basic protocol model library of OMNeT++ can be regarded as the INET Framework. INET contains models and many other protocols and components for the Internet stack. The INET Architecture is maintained for the community by the OMNeT++ team, using patches and new models submitted by community members. INET is taken as a basis by many other simulation systems and applied in unique ways, such as vehicular (Veins, CoRE), overlay / peer-to-peer (OverSim), or LTE (SimuLTE) networks. The OMNeT++ simulation kernel is standard C++ and runs basically on all platforms where a modern C++ compiler is available. The Simulation IDE requires Windows, Linux, or macOS [23].

4.4 OPNET

Optimized Network Engineering (OPNET) Tool is a licensed network simulator. It is used for simulation of both wireless and wired networks [24], riverbed technology is approved. An OPNET academic edition called OPNET IT Guru is available with a free license version for researchers and teaching groups. OPNET simulators allow users to design and study network communication, protocols, devices, applications, and routing protocols. OPNET simulators support various wireless standards such as IEEE 802.11, IEEE 802.15.1, IEEE 802.20, and satellite networks. The key benefits of the OPNET simulator are the ease of use, the user-friendly graphical user interface (GUI), and the high quality of documentation and technical support.

4.5 GlomoSim

Global Mobile Information System Simulator (GlomoSim) was developed in California, USA mainly targeted towards wireless network simulation. It was coded in Parsec and all-new protocols must be defined in Parsec as well. It also contains a front end based on java. GlomoSim has the potential to do so Run on SMP (shared-memory symmetric processor: a memory that is simultaneously accessible to all programs) and help split the program. Network in separate modules, each running as a separate operation, reduce the load on the CPU by splitting its workload. Because of this extraordinary multi-tasking feature, GlomoSim can simulate tens of thousands of nodes in single nodes simulation. GlomoSim follows the OSI layer model and supports various protocols and templates for each layer. GlomoSim has two rays and free-space radio propagation models. It was intended to support millions of nodes just as a single simulation is attributed to the technique of parallelism.

4.6 Qualnet

QualNet (Quality Networking) is a program for network evaluation and is modeled entirely as a finite state machine. It is done entirely in C++ and can run on UNIX, Windows, MAC, and Linux operating systems [25]. QualNet is related to a wide variety of libraries to simulate a variety of networks, such as Wi-Fi, sensor networks, MANET, WiMAX, etc. A solid 3D visualization tool along with a QualNet Analyzer could be used to perform the simulation. It is an efficient simulation tool that can support a 500 to 20,000 node simulation. Similar to TCP / IP, QualNet is built for testing purposes on a layered architecture consisting of a QualNet evaluation version.

Table.3 A Features Comparison of the Various Network Simulators

	NS-2	NS-3	OMNeT++	OPNET	GlomoSim	QualNet
GUI Support	Poor	Poor	Good	Excellent	Poor	Excellent
Language Support	C++ and OTCL	C++ and Python	C++	C and C++	C	Parsec C++
License Type	Open-source	Open-source	Open-source (for study and research purposes), Commercial (for industrial purposes)	Commercial	Open-source	Commercial (Separate license for academicians and others)
Supported	Linux Unix Windows	Linux Unix	Linux Unix Windows MAC	Linux	Windows, Linux, Sun	Linux Windows

Operating System	(Cygwin)	Windows	OS	Windows	SPARC Solaris	DOS
Ease of Use	Hard	Hard	Moderate	Easy	Hard	Moderate
Installation Time	Moderate	Long	Moderate	Moderate	Moderate	Short

5. VANET SIMULATORS

Drivers' reactions in different situations could have an impact on traffic flow. For example, a driver who receives a collision warning message may either hit the brake or exit the highway, depending on the distance from the accident scene. The availability of exits software that allows one to adjust the actions of vehicles (depending on the application context) is known as an integrated system or simply a VANET simulator. Just a few interconnected systems are accessible to the best of our knowledge. Mobility and network models in interconnected systems are currently being implemented in two different simulation tools. There is also a strong need for an integrated mobility simulator and a network simulator. to evaluate performance effectively.

Below, these simulators are briefly addressed as follows.

5.1 TraNS

TraNS (Traffic and Network Simulation Environment) is a simulation environment that combines both a mobility generator and a network simulator and provides a tool for creating practical VANET simulations. TraNs integrate SUMO as a mobility generator tool and NS-2 as a network simulator to provide feedback between both the vehicle activity and the mobility model. TraNS aims to avoid getting simulation results that are substantially different from those obtained by real-world experiments, as observed for current mobile ad-hoc network implementations [26]. TraNs is implemented in Java and C++ languages and can work on Linux and Windows platforms. TraNs can operate in two modes of network-centric mode and application-centric mode. There is no feedback in the network-centric mode between the SUMO mobility generator and the NS-2 network simulator.

On contrary, the mobility traces are transferred through the parser from the mobility generator to the network simulator without feedback from the network simulator. The parser converts the SUMO traces to the required format to be transferred through this parser to NS-2. As a result, TraNs is classified as a highly scalable platform that can model large-scale networks (up to three thousand vehicles) and enables Google Earth to visualize simulations.

5.2 GrooveNet

GrooveNet is an open-source hybrid simulator that combines both a mobility generator and a network simulator and enables connectivity between real and simulated vehicles. It models Inter-Vehicular Communication (IVC) through a practical street map based on topology, which facilitates protocol design and in-vehicle deployment. The simulation of thousands of vehicles in different cities in the US can be run and new models for security, networking, and vehicle interaction can be added. It offers several network interfaces and makes GPS and event-triggered (from the vehicles' on-board computer) simulations.

5.3 NCTUns

The NCTUns is an acronym of (National Chiao Tung University network simulator). Such a simulator integrates a network simulator and a mobility generator into a single module by extending the network simulator to include mobility generators and road network simulations. The Network Simulator NCTUns has many great functions and is a highly accurate network simulator and emulator capable of simulating various protocols for both wired and wireless IP networks. It could be used effectively as a simulator as it facilitates the complete connection of emulation and simulation. And, it uses the Linux TCP / IP protocol stack to produce high fidelity simulation data. It can run any real-life UNIX program on a simulated node without any alteration. NCTUns is a simulator based on C++ that works with Linux Fedora. The GUI software consists of four main elements. They are the "Packet Animation Player", "Performance Monitor", "Node Editor," and "Topology Editor".

5.4 Veins

Veins (Vehicular In-Network Simulations) is an open-source framework for running vehicular network simulations. It is based on two well-established simulators: OMNeT++, an event-based network simulator, and SUMO a road traffic simulator. It extends these to offer a comprehensive suite of models for Inter-Vehicular Communication (IVC) simulation [27]. This encourages researchers to use it and publish articles on Veins, which helps to raise the user numbers. Veins is available for download free of charge. The mobility generator SUMO is running in parallel and it includes a large number of simulation models for vehicle networks, executed by the event-driven network OMNeT++ simulator.

During the simulation of SUMO, the vehicle movements are expressed directly as the node movement in the OMNeT++ network simulator. In OMNeT++ simulation, veins use a TCP link and Python scripts to allow SUMO to act as a mobility model, and nodes can communicate freely with the simulation of running road traffic.

5.5 MobiREAL

MobiREAL is a realistic network simulator for MANET. It provides a new methodology to model and simulate realistic mobility of nodes and enables to evaluate MANET applications in more actual environments [28]. It is a new mobile-device network simulator for all-embracing society. It can simulate the realistic mobility of humans and vehicles and adjust their actions in the sense of an application.

However, using such a simulator can provide more in-depth network application performance evaluation, routing protocols, and infrastructure which are not easily evaluated by current simulators. MobiREAL can easily illustrate C++ node mobility. It adopts a probabilistic rule-based model to explain mobile node actions, which are also used in human behavior cognitive modeling. The model proposed helps the researchers to define, on a basis of their location, their surroundings (obstructions and neighboring nodes), how mobile nodes adjust their destinations, roads, and speeds, the information collected from apps, etc.

Thereby, MobiREAL can simulate the mobile ad-hoc network by adding mobility support facilities to the Georgia Tech Network Simulator (GTNetS), which newly created mobile nodes join/leave, move, and packet collision.

Table.4 A Features Comparison of the Various VANET Simulators

	TraNS	**GrooveNet**	**NCTUns**	**Veins**	**MobiREAL**
GUI	Yes	Yes	Yes	Yes	Yes
Ease of setup	Moderate	Moderate	Hard	Easy	Easy
Network simulator	NS-2	-	-	OMNeT++	based on GTNetS
Mobility generator	SUMO	GrooveNet	NCTUns	SUMO	MobiReal
Ease of use	Moderate	Hard	Hard	Moderate	Hard

6. PROPOSED NETWORK MODEL

6.1 Network Settings

To demonstrate the performance of the vehicular network, the VANET environment is considered as follows:

1. The environment is the Basrah City, in Urban streets using the real-time integration of OMNeT++ and SUMO.
2. The vehicle speed is 40Km/h and 60Km/h without considering any obstacles or disorder in signal transmission over a wireless channel during driving.
3. The vehicle density is represented by a number of nodes (vehicles) range that varies between 10-40 ignoring the sudden change in vehicle direction.
4. In OMNeT++ simulation, Veins will use a TCP link and Python scripts to allow SUMO to act as a mobility model. Thereby, the nodes can communicate freely with the simulation of running road traffic. Table 5 illustrates the simulation parameters which are considered in our network.

Table 5. Simulation Parameter

Parameter	**Value or Protocol**
OMNeT++ Version	OMNeT++ V 5.5.1
SUMO Version	SUMO 1.6.0
INET Version	INET 4.2.1
Veins Version	Veins 5.0
Vehicles Number	10,20,30,40
Speed	40Km/h, 60Km/h
Simulation Time	600 Seconds
MAC Protocol	IEEE802.11p

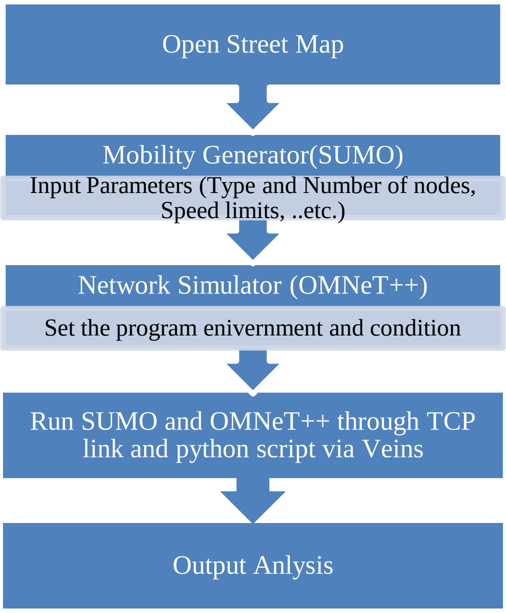

Open Street Map
Mobility Generator(SUMO)
Input Parameters (Type and Number of nodes, Speed limits, ..etc.)
Network Simulator (OMNeT++)
Set the program enivernment and condition
Run SUMO and OMNeT++ through TCP link and python script via Veins
Output Anlysis

Fig 3: A Typical Design of a Network Simulator in OMNeT++ and SUMO for VANET Environment

6.2 Simulation Results

In this section, the proposed network model is implemented using OMNeT++ and SUMO with Veins and INET frameworks. The OMNeT++ is a network simulator that works together with SUMO which is a mobility generator. The mobility generator simulates the actual traffic simulation of the system's built vehicles, and the mobility generator output is the trace file of mobility patterns in real-life maps of vehicles. The OMNeT++ network simulator receives mobility traces as input to its simulation and converts the real-world movement of the vehicle into the network node movement. The network simulator will then run its simulation and the programmer will be able to analyze the fully configured networks and test the network parameters that are targeted by the simulation of the network.

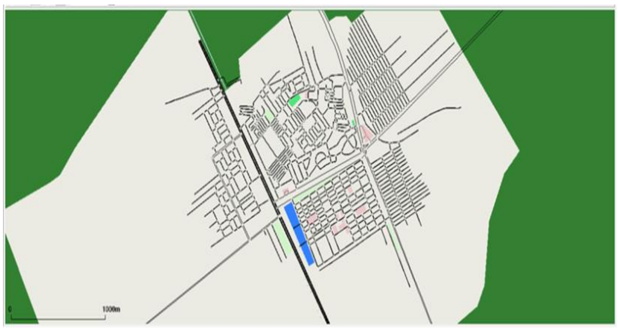
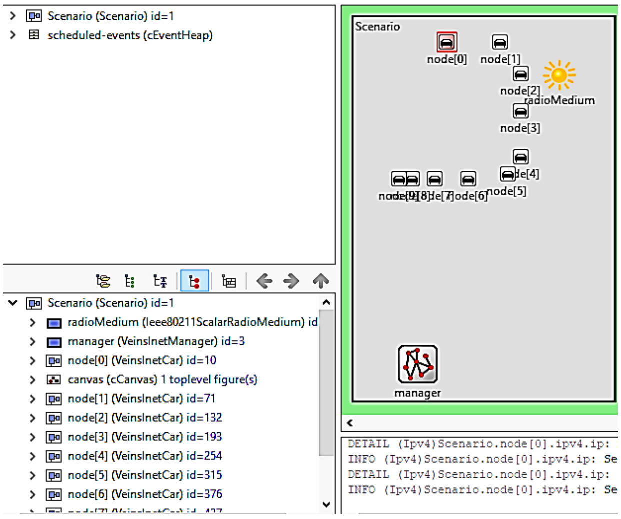


Fig 4: Basrah streets in SUMO 1.6.0


Scenario (Scenario) id=1
scheduled-events (cEventHeap)

Scenario (Scenario) id=1
radioMedium (Ieee80211ScalarRadioMedium) id
manager (VeinsInetManager) id=3
node[0] (VeinsInetCar) id=10
canvas (cCanvas) 1 toplevel figure(s)
node[1] (VeinsInetCar) id=71
node[2] (VeinsInetCar) id=132
node[3] (VeinsInetCar) id=193
node[4] (VeinsInetCar) id=254
node[5] (VeinsInetCar) id=315
node[6] (VeinsInetCar) id=376

```
DETAIL (Ipv4)Scenario.node[0].ipv4.ip:
INFO (Ipv4)Scenario.node[0].ipv4.ip: Se
DETAIL (Ipv4)Scenario.node[0].ipv4.ip:
INFO (Ipv4)Scenario.node[0].ipv4.ip: Se
```

Fig 5: Simulation using OMNeT++

Thereby, the network performance in terms of the packet delivery ratio and network throughput are examined using a different number of nodes and two-speed ranges 40Km/h and 60Km/h, as shown in Figure 3. The Packet Delivery Ratio is shown better in speed 40Km/h than 60Km/h, and the Packet Delivery Ratio is computed as follows:

$$\text{Packet Delivery Ratio} = \frac{\mathit{Total\ No.of\ Packet\ Recived}}{\mathit{Total\ No.of\ Packet\ Sent}}$$

Meanwhile, the network throughput increases as the number of nodes (vehicles) increases, but it would decrease as the vehicle speed arises up to 60Km/h. That's means due to the reduction in the packets delivery at the destination. Throughput is computed as,

$$\text{Throughput} = \frac{\mathit{Total\ Data\ Sent}}{\mathit{Total\ Time}}$$

7. CONCLUSION

Researchers have built precise and practical simulation instruments due to their growing popularity and attention in VANET. In this paper, a brief survey has been introduced on many publicly accessible mobility generators, network simulators, and the VANET simulator. A mobility generator is built to provide a network simulator with realistic traces of vehicle movements replicated in a realistic environment. The traces generated by the mobility generator shall be translated into the network simulator as the movement of the nodes within the networks. The choice of the most appropriate full VANET network ad-hoc vehicle simulator is dependent on a variety of factors; the application for which the network is built, the functionality of each simulation tool, and the financial factors that influence the use of commercial or open-source software. Basic simulation execution is achieved via SUMO's integration of the mobility generator with the network simulator OMNeT++ and the integrated Veins framework. . As a result, the proposed network model in this paper was created via an overall network simulator design perspective followed by full-scenario empirical results to demonstrate how the mobility generator and network simulator integration is performed.

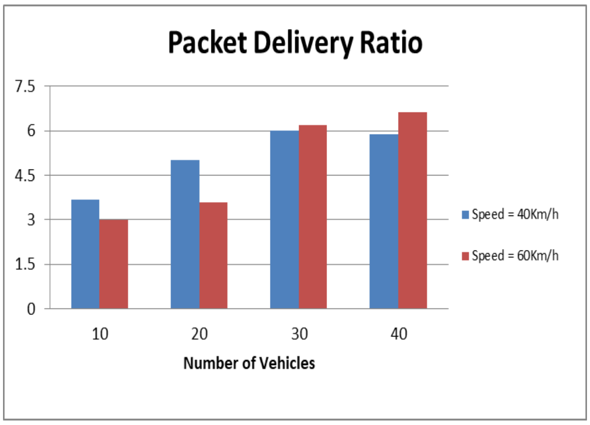


Fig 6: Packet delivery Ratio vs. Number of Vehicles

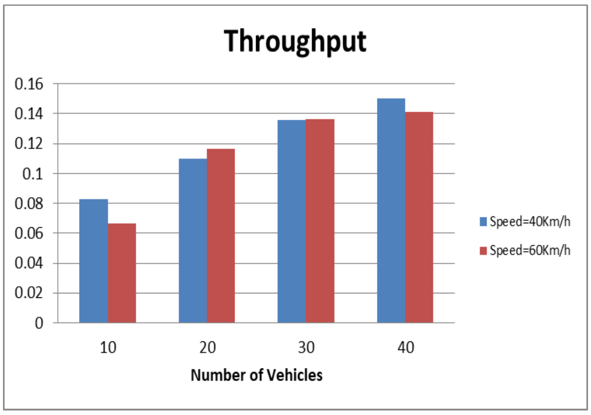


Fig 7: Throughput vs. Number of Vehicles

8. REFERENCES

[1] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5G for Vehicular Communications", IEEE Communications Magazine, no. 1, pp.111-117, 2018.

[2] W. B. Jaballah, M. Conti, and C. Lal. "A Survey on Software-Defined VANETs: Benefits, Challenges, and Future Directions", arXiv:1904.04577 [cs.NI], May 2019.

[3] B. Qian, H. Zhou, T. Ma, Y. Xu, K. Yu, X. Shen, and F. Hou, "Leveraging Dynamic Stackelberg Pricing Game for Multi-Mode Spectrum Sharing in 5G-VANET", IEEE

Transactions on Vehicular Technology, vol. 69, no. 6, 2020.

[4] S. M. Hatim, S. J. Elias, N. Awang, and M. Y. Darus, "VANETs and Internet of Things (IoT): A Discussion", Indonesian Journal of Electrical Engineering and Computer Science, vol. 12, no. 1, pp. 218-224, Oct. 2018.

[5] A. V. Ostroukh, and H. Elhadi. "Comparative Study of Routing Protocols in Vehicular Ad-hoc Networks (VANETS)", International Journal of Advanced Studies (iJAS), vol. 4, no. 2 pp. 9-14, 2014.

[6] K. Tanuja, T.M. Sushma, M. Bharathi, and K. H. Arun, "A Survey on VANET Technologies", International Journal of Computer Applications, vol. 121, no. 18, July 2015.

[7] R. Kumar, and M. Dave. "A Comparative Study of Various Routing Protocols in VANET", International Journal of Computer Science Issues (IJCSI), vol. 8, issue 4, no. 1, July 2011.

[8] N. Upadhyaya and J. S. Shah, "AODV Routing Protocol Implementation in Vanet", International Journal of Advanced Research in Engineering and Technology, vol. 10, no. 2, pp. 585-595, 2019.

[9] C. Tripp-Barba, A. Zaldívar-Colado, L. Urquiza-Aguiar, and J. A. Aguilar-Calderón, "Survey on Routing Protocols for Vehicular Ad-hoc Networks Based on Multimetrics", Electronics, vol. 8, no. 10, 2019.

[10] S. Malik, and P. K. Sahu, "A Comparative Study on Routing Protocols for VANETs", Heliyon, vol. 5, no. 8, August 2019.

[11] F. J. Martinez, C. K. Toh, J. C. Cano, C. T. Calafate, and P. Manzoni. "A Survey and Comparative Study of Simulators for Vehicular Ad-hoc Networks (VANETs)", Wireless Communications and Mobile Computing, vol. 11, no. 7, pp. 813-828, 2011.

[12] F.D.Cunha, L. Villas, A. Boukerche, G. Maia, A. C. Viana, R. A. F.Mini, and A. A. F. Loureiro, "Data Communication in VANETs: Survey, Applications and Challenges", Ad Hoc Networks, pp.90-103, 2016.

[13] J. A. Junior, and E. C.G. Wille. "Routing in Vehicular Ad Hoc Networks: Main Characteristics and Tendencies", Journal of Computer Networks and Communications, 2018.

[14] M. A. Elgazzar, A. Alshareef, "VANET Simulator: Full Design Architecture", International Journal of Engineering and Advanced Technology (IJEAT), vol. 9. no. 3, Feb 2020.

[15] SUMO https://www.eclipse.org/sumo/

[16] V. Nampally, M. R. Sharma, and K. R. Balaji, "Simulators for VANET", International Journal for Research in Applied Science & Engineering Technology (IJRASET), vol. 5, no. IX, Sept. 2017.

[17] NS-3 https://www.nsnam.org/

[18] M. N. Mittal, and S. Choudhary, "Comparative Study of Simulators for Vehicular Ad-hoc Networks (Vanets)", International Journal of Emerging Technology and Advanced Engineering, vol. 4, no. 4, pp.528-537, April 2014.

[19] S. ur Rehman, M. A. Khan, T. A. Zia, and L. Zheng, "Vehicular ad-hoc networks (VANETs)-An overview and Challenges", Journal of Wireless Networking and Communications, vol. 3, no. 3, pp. 29-38, 2013.

[20] O. Pattnaik and B. K. Pattanaya, "Performance Analysis of MANET and VANET based on Throughput Parameter", International Journal of Applied Engineering, vol. 12, no. 18, pp. 7435-7441, 2017.

[21] B. Sahadev, K. S. Zade, and S. H. Sheikh, "Survey on Realistic Simulation for Comparison of Network Routing Protocol in VANET", In 2nd National Conference on Innovative Paradigms in Engineering & Technology, pp. 26-29. 2013.

[22] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, "A Comprehensive Survey on Vehicular Ad hoc Network", Journal of network and computer applications, vol. 37, pp. 380-392, Jan. 2014.

[23] OMNeT++ https://omnetpp.org/

[24] T. Arvind, "A Comparative Study of Various Network Simulation Tools", International Journal of Computer Science & Engineering Technology (IJCSET), vol. 7, no. 8, pp.374-378, Aug 2016.

[25] M. H. Kabir, S. Islam, M. J. Hossain, and S. Hossain. "Detail Comparison of Network Simulators", International Journal of Scientific & Engineering Research, vol. 5, no. 10, pp. 203-218, Oct. 2014.

[26] L. RAJA, "Study of Various Network Simulators", International Research Journal of Engineering and Technology (IRJET), vol. 5, no. 12, Dec 2018.

[27] https://veins.car2x.org/

[28] http://mobireal.net/