Public Safety Application for Approaching Emergency Vehicle Alert and Accident Reporting in VANETs Using WAVE

by

Harshad Phule

A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science
in Computer Engineering

Supervised by

Assistant Professor Dr. Andres Kwasinski
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
May 2012

Approved by:

Dr. Andres Kwasinski, Assistant Professor
Thesis Advisor, Department of Computer Engineering

Dr. Dhireesha Kudithipudi, Assistant Professor
Committee Member, Department of Computer Engineering

Mr. Joseph Milhorn, Systems Engineer-IV, Harris Corporation
Committee Member, Department of Public Safety and Private Communications

Dedication

This thesis is dedicated to my finacée, my parents, my family and my friends. Their constant support and encouragement played a big role to help me finish.

Acknowledgments

I am grateful for the guidance and support provided by the thesis committee throughout the process. Also, I am grateful to Christoph Sommer, Matt Prokop and Brent Epperly for their help and support.

Abstract

Today's transport system has evolved from horse driven carriages and paved roads to a more complex road transport system made up of a variety of vehicles and other infrastructure, all put in place in order to support safe and efficient mobility of vehicles. The next step to further improve the transportation system of today is to make the vehicles and roadside infrastructure more intelligent by making them communicate with each other. This new ability will help find new solutions to current problems like traffic congestion, vehicular collisions, monitoring of adherence to traffic rules and alerting the drivers of hazardous conditions. Vehicular Ad-Hoc Networks (VANETs) are vehicle to vehicle and vehicle to road side infrastructure networks which make this possible by providing support to numerous applications aimed towards improving safety and driving experience on the road.

A Public Safety VANET application adhering to the IEEE Wireless Access in Vehicular Environments (WAVE) standard and requirements laid down by National Highway Traffic Safety Administration (NHTSA) was designed, implemented, simulated and tested over a scalable open-sourced test bed aimed towards simulating a complete Intelligent Transportation System consisting of various application operating together. The application supports two features:

Approaching Emergency Vehicle Alert: An Approaching Public Safety Vehicle (Police/Fire/EMS) in a state of emergency will alert the vehicles in its path using the VANET, causing them to give way.

Post-Crash Warning: A vehicle involved in a crash will immediately alert approaching vehicles about its current state using the VANET, helping them come to a halt at a safe distance, hence avoiding pileups.

The performance and adherence to the requirements was analyzed, and a demonstration of the system was prepared to showcase the application in action.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

As humans evolved, they have always had the need to travel great distances. Over many centuries, humans being intelligent beings have always strived to invent better ways to achieve this and also developed and improvised their environment to better support and facilitate this. The Egyptian civilization is a good example, where the horse driven carriages and paved roads formed one of the first road transport systems. This over time has evolved to what we now have today, a more complex road transport system made up of a variety of vehicles, paved roads, bridges, tunnels, clover leaf interchanges, traffic signals and other infrastructure, all put in place in order to support the mobility of vehicles.

The next step to further improve the transportation system of today is to make the vehicles and roadside infrastructure more intelligent by devising a means allowing them to communicate with each other. This new ability will help find new solutions to current problems like traffic congestion, vehicular collisions, monitoring of adherence to traffic rules and alerting the drivers of hazardous conditions. Ensuring the safety of the lives of people has become a major area of concern as vehicle density is on the rise and vehicles are now capable of travelling faster than ever before.

Most modern cars of today already have intra-vehicular networks which allow seamless wireless connectivity between the vehicle's electronic control center and other electronic devices like smart phones, Global Positioning Systems, Bluetooth headsets and media players. But a network between vehicles is still not available. Vehicular Ad-Hoc Networks (VANETs) are aimed towards doing just this. VANETs are emerging networks among vehicles and roadside infrastructure to serve applications which will improve safety on road networks and also provide content delivery on the move. VANETs are an extreme case of Mobile Ad-Hoc Networks (MANETs) characterized by major variations in node density, network topology, environment and high speed mobility. The IEEE has developed a system architecture called WAVE (Wireless Access in Vehicular Environments), which is a conjunction of the standards IEEE 802.11p and IEEE 1609.x to address these characteristics unique to VANETs and provide wireless access in vehicular environments. It is engineered to connect wireless devices attempting to communicate in a rapidly changing networking environment and in situations where data transactions should be done in shorter amounts of time. Research in this area has gained considerable momentum over the last decade with several research projects around the world: Vehicles Safety Communications Consortium [4] (USA), Internet ITS Consortium [1] (Japan), Network on Wheels project [2] (Germany) and PReVENT project [3] (Europe).

1.1 Motivation

VANETs open up a whole new class of applications aimed towards providing an improved, safer and co-operative driving experience. Here, according to [24], applications can be further divided into two types:*Comfort Applications*, which cater towards improving the driver's experience and *Safety Applications* which help improve safety of the drivers.

The big picture of an Intelligent Transportation System would consist of not just VANETs, but many underlying networks connected to each other in order to support various features as shown in Figure 1.1. The vehicles will connect to other vehicles and stationary Road Side Units (RSUs). The Road Side Units, Traffic Signals, Toll Collection Centers will be part of a backbone network which also connects various control centers like Emergency Dispatch Centers, Traffic Control Rooms and Police Control Rooms together forming one seamless network.

Public safety applications like approaching emergency vehicle warning, post-crash warning, accident reporting, blind merge warning and pre-crash sensing are of greater interest as they make the most meaningful use of

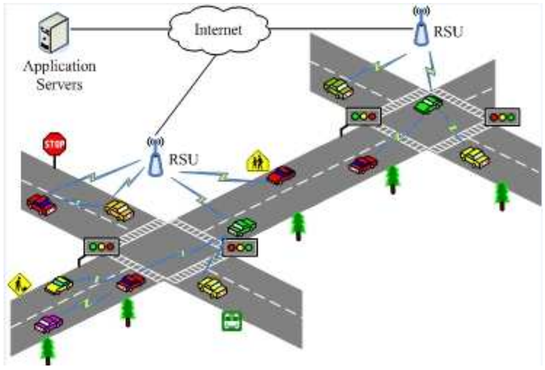

Figure 1.1: Intelligent Transportation System [23]

VANETs and do justice to the expenditure and infrastructure overhaul needed to support them. The National Highway Traffic Safety Administration (NHTSA) has identified and laid down requirements for Intelligent Vehicle Safety Applications in [4].

The above mentioned applications and many more will have to co-exist on the same network and utilize shared resources in order to provide the user with all the promised features. Testing the performance of applications and other related features of VANETs is necessary when in the development phase. The complex behavior of vehicles, drivers and the different patterns observed in the real world are difficult to replicate due to constraints on available resources making it difficult to recreate VANETs in real world for testing purposes. For example, if an application was to be tested in a downtown setting with 50 cars, one would actually need 50 cars fitted with the test equipment, 50 people to drive the car, and a testing ground as well. Real world implementations are observed in work done by Cottingham, David *et al.* [10] where the performance of IEEE 802.11a in a vehicular context was evaluated by using a wireless access point as a roadside unit, a laptop with IEEE 802.11a wireless network interface card and a GPS unit placed in a car as the mobile node. Shen, Yong, *et al.* [15] evaluated performance of IEEE 802.11a, 8011.b, 802.11g wireless channels in vehicular environments using two laptops placed in two cars with GPS systems with limited scenarios like *line of sight*- where the two cars pass each other in opposite directions and *obstacle*- where a stationary van acts as an obstacle between the two cars. In [9] Pierpaolo Bergamo *et al.*used two cars with laptops and omnidirectional antennas to test the performance of IEEE 802.11b wireless networks in aggressive mobility scenarios. In [16], Singh J.P *et al.*used two laptops running linux, GPS units and cars to test WLAN Performance under varied stress conditions in Vehicular Traffic Scenarios. It was observed that in such implementations, the number of vehicles is limited, and only a few situations could be successfully recreated making it difficult to test an application that requires a large number of vehicles to be present in the test scenario.

The other option available to the research community is to simulate the VANETs using Network Simulators with defined mobility models to simulate the movement of the nodes, or use a dedicated Traffic Simulator to achieve the same.

In order to fully test an Application, it needs to be tested in a realistically simulated environment with interoperability between all the different networks. This can be achieved by a simulation test bed which

- Realistically models the movement and behavior of vehicles.
- Models VANETs and all the supporting networks and the protocols used.
- Provides the flexibility to program and add new applications.

1.2 Thesis Statement

The goal of this thesis was to gather requirements, design, implement and test a Public Safety Application for VANETs which conforms to the WAVE standard using a scalable test bed. The application supports the following features:

Approaching Emergency Vehicle Alert: Today, Police, Fire and Emergency Services vehicles alert the surrounding on-road vehicles about the state of their emergency using audio-visual alerts : flashing lights and sirens. The existence of VANETs is exploited here to disseminate this information using a third medium, straight to the vehicles. Also, information like direction, distance and route is passed. This helps the driver of the recepient vehicle make an informed decision about whether and when he/she needs to pull over to give way to the Public Safety Vehicle well in advance. This facilitates enhanced co-operation between first responders and other vehicles on the road.

The simulation also models the reaction of human drivers to such an application by making the vehicles pull over, give way to the Public Safety Vehicle and then continue on their way, hence actively modifying the overall flow of traffic in the simulation test bed.

Accident Reporting (Post-Crash Warning) : Most modern vehicles today are equipped with a variety of sensors on-board (Anti-lock Brakes, Air-bag deployment sensors, accelerometers, gyroscopes) which when used together can detect a collision. Vehicles involved in an accident will use the VANET to immediately start sending out messages to surrounding vehicles alerting them about the crash. Information about the crash will also be transmitted. This will alert approaching vehicles that are on the path of the crashed vehicle to slow down and stop at a safe distance, hence avoiding a pile up and saving more people from injuries and loss of life, especially during bad weather conditions with low visibility. The simulation will again model the reaction of human drivers to this feature of the application by making the vehicles stop at a safe distance away from the crashed vehicle, hence altering the flow of traffic in the simulation test bed.

This Master's thesis will provide a good starting point towards achieving total simulation of an Intelligent Transportation System, and will also provide a platform to research other aspects of VANETs.

Chapter 2

Related Work

The research community has been very proactive in the recent years with ongoing research regarding many aspects of VANETs. One of the major breakthroughs was the IEEE 802.11p wireless Standard, an approved amendment to the IEEE 802.11 standard. This section will provide an in-depth overview about VANETs, the WAVE (Wireless Access In Vehicular Environment) architecture, Application requirements laid down by NHTSA (National Highway Traffic Safety Association) and VEINS (Vehicles in Network Simulation)

2.1 Vehicular Ad-Hoc Networks:

Vehicular Ad-Hoc Network can be defined as a highly mobile Ad-Hoc Network between vehicles, road side units, and other traffic control related infrastructure in which all the mobility is accounted for by the vehicles. VANETs are characterized by the following:

- *Highly dynamic network topology*: Vehicles, depending on their position and route will constantly be moving from one VANET to another.
- *Varying node density*: Busy downtown areas during peak hours will result in high node density whereas rural areas will have a lower node density.
- *Varying node velocities*: The relative speed of two vehicles moving in an opposite direction on a highway will be considerably high compared to two vehicles just following each other, at almost the same speed.

- *Varying environment*: A VANET in a downtown environment will have many manmade structures leading to poor connectivity between the nodes. Topographical changes and other obstacles too will hinder with the connectivity in a rural setting.

2.2 WAVE: Wireless Access in Vehicular Environment

WAVE is a system architecture developed by the IEEE aimed towards facilitating communication between vehicles and road side infrastructure. It was developed as the existing IEEE 802.11a,b,g standards would not work efficiently with tasks like authentication and update of routing tables with the ever changing network topology and short amount of time available for conection setup and communication.

2.2.1 Brief History of WAVE [22]

1. The Intermodal Surface Transportation Efficiency Act of 1991 (ISTEA) mandated the creation of Intelligent Vehicle Highway Systems (IVHS), a program aimed towards tackling congestion, pollution, safety and conserve fossil fuels. The US Department of Transportation (DOT) was handed the responsibility of this program.

2. In 1999, FCC granted 75MHz of spectrum in the range of 5.85 - 5.925GHz range for IVHS.

3. In 2004 IEEE Task Group 'p' started developing an amendment to the 802.11 standard to include support for vehicular environments - IEEE 802.11p. IEEE work group 1609 developed the following protocols:

    - IEEE 1609.1
    - IEEE 1609.2
    - IEEE 1609.3
    - IEEE 1609.4

Together, IEEE 802.11p and IEEE 1609.x formed the WAVE architecture as shown in Figure 2.1.

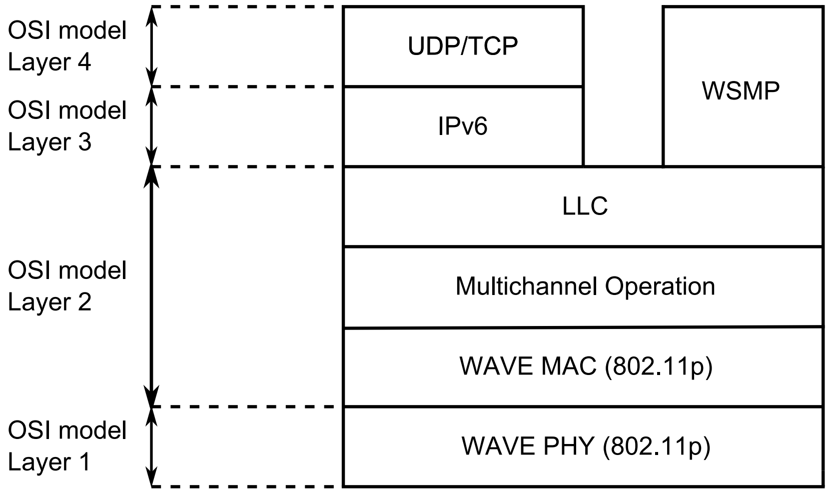

Figure 2.1: WAVE Communications Protocol Stack [6]

2.2.2 IEEE 802.11p: WAVE PHY and MAC

The Physical and Medium Access Control layers of the WAVE architecture are defined by the IEEE 802.11p standard. It is based on the 802.11 standard as it is a widely known well implemented standard in the industry with many manufacturers guaranteeing interoperability. The major imporovements observed in the 802.11p standard are:

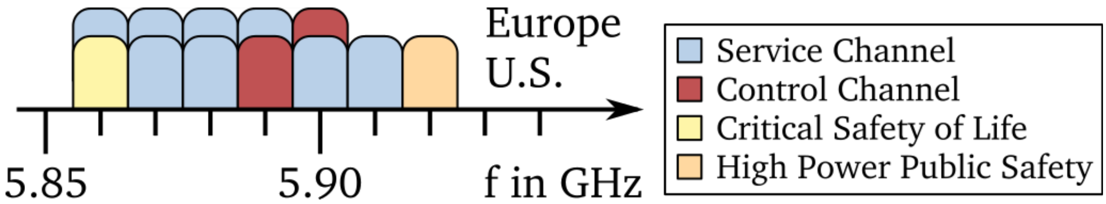

Figure 2.2: Available Channels in 802.11p [18]

Fast Connection setup: As VANETs usually have a highly dynamic network topology, the amount of time for which two vehicles are in communication range is less. The worst case is when two vehicles are travelling in opposite direction at a high speed. Faster connection setup ensures there is enough time available to communicate the data while the connection lasts.

10-Mhz channels: Figure 2.2 shows the ditribution of the allocated 75MHz spectrum into 10 Mhz control and service channels. Europe will support one Control Channel and four Service Channels. The Control Channel is used for communicating Wave Short Messages and to announce WAVE services. The Service Channels are used for application interactions. In the US, there will be support for two additional channels, Critical Safety of Life - to be used by SOS applications and a High Power Public Safety channel to be used by Public Safety applications ensuring a better range due to the high transmit power being used.

Random MAC address: To enure the anonymity of the vehicles, and in order to make them untraceable, support for a random MAC address is added. This makes it difficult to associate data collected over the VANET to one particular vehicle and to track the vehicle using this data.

Priority control: This helps achieve Quality of Service (QoS) in VANET applications. Data related to differet applications will be prioritized providing critical data access to the channel.

Power control: This feature enabled the applications to specify the transmit power used to trasmit each packet over the channel. This is useful for having intelligent transmit power management by the application depending on the surrounding topographical charecteristics.

Halved Data Rates: The data rates supported by 802.11p are half of those supported by 802.11a. This helps reduce errors in transmission in a highly mobile environment. Also, 16 QAM is used in mobile environments.

Table 2.1 shows comparison between IEEE 802.11a and IEEE 802.11p PHY values.The main observable difference here is that the energy per symbol in 802.11p has been doubled and the data rate has been halved.

Table 2.1: Comparison between 802.11a and 802.11p PHY values [21]

Parameter	**802.11p**	**802.11a**
Channel bandwidth	3 to 27 Mbps	6 to 54 Mbps
Data rates	10 MHz	20 MHz
Slot time	16μs	9 μs
SIFS time	32 μs	16 μs
Preamble length	32 μs	20 μs
PLCP header length	8 μs	4 μs
Air Propagation time	$<$4 μs	$<<$1 μs
CWmin	15	15
CWmax	1023	1023

2.2.3 IEEE 1609.4: Multichannel Operation [5]

The IEEE 1609.4 standard helps support the multichannel operation feature. The physical channel is time division multiplexed to be used as a control channel and a service channel. The standard adds support for the following features:

Control Channel Monitoring: All WAVE devices have to monitor the Control Channel during a specific interval called the Control Channel Interval for public safety related messages and priavate service advertisements for the services running on the other Service Channels. The IEEE 1609.4 handles the switching between Control Channel and Service Channels.

Synchronized Switching: Devices that have only a single channel available for communication cannot monitor the Control Channel and the Service Channel at the same time. For such devices to communicate effectively, there is need for communicating WAVE devices to be synchronized so that they all monitor the Conrol Channel during the same time interval and data transfer on the Service Channels occurs in the consequent time interval. This standard provides the guidelines to attain synchronization with reference to an absolute external time reference - Coordinated Universal Time (UTC), commonly provided by a Global Positioning System (GPS).

Channel Routing: The routing of data received from the higher layers to the right channel - Control Channel or one of the Service Channels is done here. Also, the data received from the lower layers is routed to the correct networking protocol at the higher layer - IPv6 or Wave Short Message Protocol (WSMP).

2.2.4 IEEE 1609.3: Wave Short Message Protocol (WSMP) [6]

The WAVE architecture supports UDP/TCP at the Transport layer and IPV6 at the Networking layer respectively. Apart from providing this support, it provides us with a new protocol - WSMP, aimed towards VANETs which absorbs both Transport (OSI Layer 4) and Network (OSI Layer 3) layer functionality into a single protocol called the Wave Short Message Protocol. The IEEE 1609.3 standard defines the WSMP. It provides networking services to accept data from higher layers for transmission over WSMP and delivers received WSM data to higher layers.
The main goal of WSMP is to support high-rate, low-latency communications between WAVE devices in a rapidly varying radio frequency environment. WSMP data is the only kind of data allowed to be transmitted on the Control Channel. All IPV6 data is restricted to the Service Channels. IPV4 is not supported by WAVE.

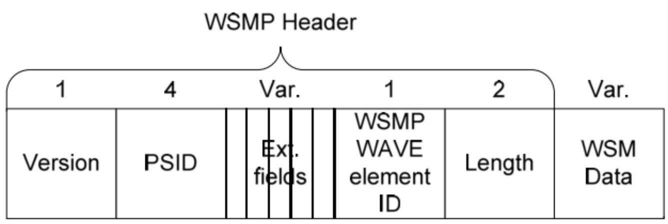

Figure 2.3: Wave Short Message [6]

Figure 2.3 shows the building of a WSM package. The fields that are important and which are used are explained below:

Version: The WSMP version indicates the version of the WSM protocol. The current version number is 2. Any WSMP packet received with a version mismatch is discarded.

PSID (Provider Service Identifier) : The PSID is used to determine the appropriate higher layer destination of the WSM. It is specified as a variable-length ordered sequence of octets.

WSMP Header Extension Fields: Extension fields are used to carry information in the WSMP header. Information like Channel Number, Data Rate, Transmit Power used and Two Dimensional location can be transmitted as part of the header for use by the recipient. The location information may be used for routing as well.

WSMP WAVE element ID: This ID indicates the type of WSM that follows the header.

Length: Indicates the length of the following data field in octets.

WSM Data: Contains the higher layer information being transmitted.

When using WSMP to transmit data, the application can specify the following to the lower layer:

- Peer MAC address: MAC Address of the recipient.
- User Priority: Priority assigned to this message, for use by the lower layers.
- Channel Number: Whether this message should be transmitted on the Control Channel or a Specific Service Channel.
- Data rate: The Data Rate at which this message should be transmitted.
- Transmit Power Level: The Transmit power to be used when transmitting this message.

2.3 Simulating VANETs

Two major trends have been observed in the research community when it comes to simulating VANETS. The two are explained below:

2.3.1 Stand-alone Network Simulator

In this approach, a network simulator is used to simulate the networking aspect of the VANET and a mobility framework within the network simulator, coupled with a mobility model is used to model the mobility of the vehicles in the VANET. Figure 2.4 shows the types of mobility models used.

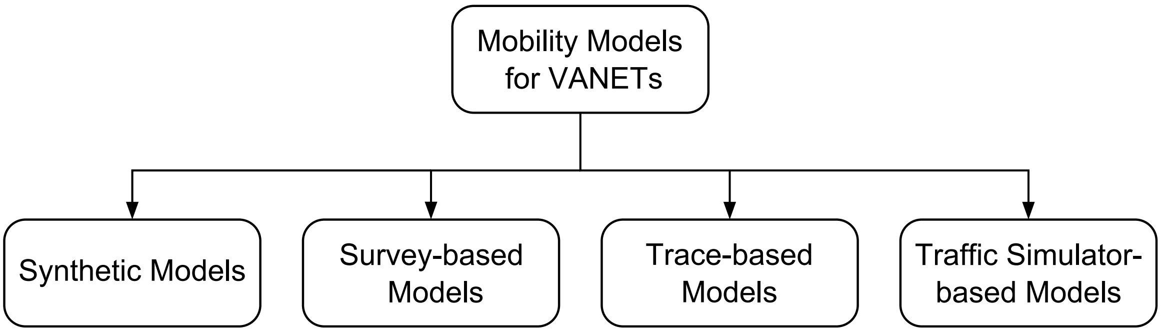

Figure 2.4: Types of Mobility Models [14]

Synthetic Models: These are mathematical models which model the flow of traffic as stochastic models - where the vehicle's mobility is considered to be purely random, traffic stream models - where the mobility of vehicles on a road is considered to be a hydrodynamic phenomenon, and Car following models - where the behavior of each vehicle is modeled depending on the vehicles ahead of it. The major drawback of such models is the complexity involved in modeling detailed human behaviors. In [**?**] W.Hsu *et al.*discuss the Weighted Waypoint Mobility Model.

Survey Based Models: Use information on flow of vehicles gathered from surveys. For example, surveys on US workers' commuting time, lunch time, travelling distance provided by US department of Labor are used to generate a mobility model in an attempt to realistically model the traffic at different times of the day. The agenda-based mobility model [**?**] is a good example as it combines social activities and geographic movements, and movement of each node is based on the driver's agenda for that time of the day.

Trace Based Models: Vehicle traces are created by directly extracting generic mobility patterns from vehicle movement traces. Though this approach gives a rather realistic mobility model, it has a major drawback of limited availability of vehicular traces.

Examples:

- Network Simulator -2 with freeway mobility model was used in [7] for performance evaluation.
- Qualnet, a network simulator with the VanetMobiSim tool add-on was used in [12] for performance evaluation of IEEE 1609 WAVE and IEEE 802.11p for vehicular communications.

2.3.2 Network Simulator Coupled with Traffic Simulator

In this method of simulating VANETS, a network simulator is coupled with a dedicated traffic simulator. Here the network simulators are only responsible for simulating the networking protocols used by the VANET. The mobility of each node in the simulation is controlled by the dedicated Traffic Simulator and the updated position is sent to the Network Simulator periodically.

In [17], Christoph Sommer *et al.*realized the need for a robust simulation platform for VANETS wherein the results of the simulation are not entirely dependent and heavily influenced by the mobility model in use. This lead to the development of VEINS (Vehicles in Network Simulation) [17], an open source simulation framework which makes use of two simulators- OMNeT++ for handling the network simulation and SUMO for handling microscopic road traffic simulation. The two simulators run in parallel and communicate over a TCP connection as shown in Figure 2.5. Traffic Control

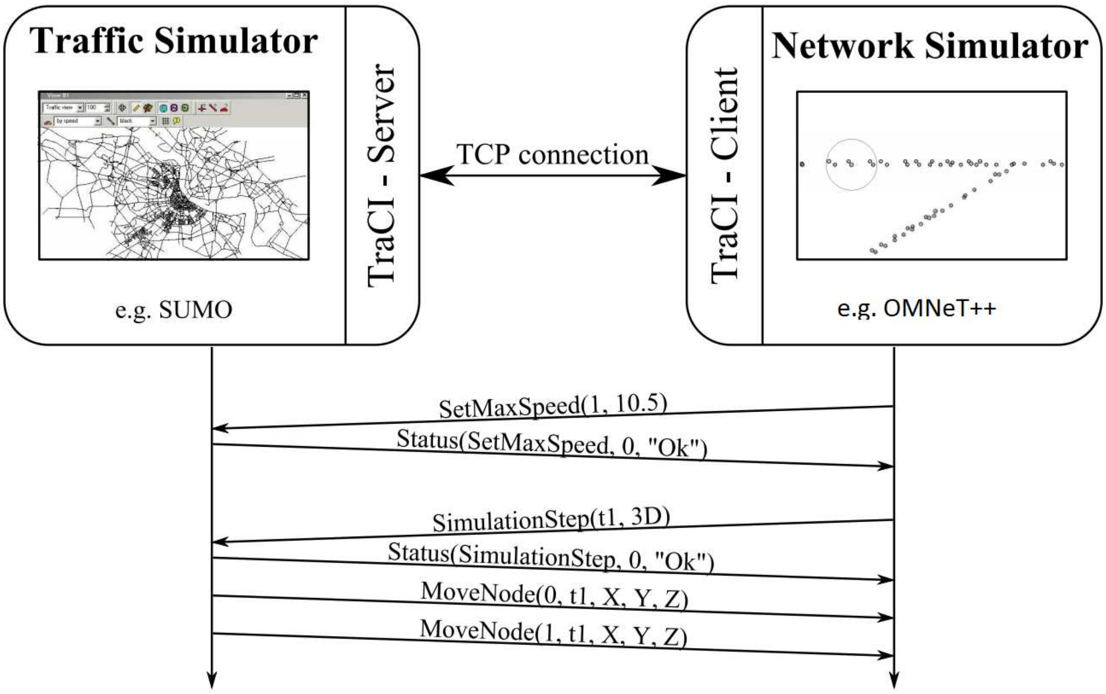

Figure 2.5: Network Simulator OMNeT++ coupled with Traffic simulator SUMO using TraCI [11]

Interface (TraCI), a standardized protocol is used for communication. This approach towards simulating VANETs provides the following advantages:

- Network Simulation is independent of Traffic Simulation.
- Traffic is simulated more realistically, complete with traffic signals, lanes, junctions and speed limits.
- Network Simulator can also wield control over the vehicle and alter associated properties, and provide feedback on the mobility model used, making it more realistic and dynamic.

2.3.3 Obstacles

The biggest challenge that VANETs face in urban scenarios is the presence of buildings and other manmade structures which hinder with the wireless connectivity due to shadowing and multipath fading effects. Shadowing is defined as the fluctuations in received signal power due to objects obstructing the path between the communicating devices. Multipath induced fading prevails in urban scenarios. It is very important that obstacles be modeled in the simulation in order to get accurate results, and to ensure that special measures are taken in order to overcome this challenge. In [19], Christoph sommer *et al.*present a realistic and computationally inexpensive simulation model for IEEE 802.11p radio shadowing in urban environments. It is an emperical model based on real world measurements. It is computationally inexpensive as it does not take the usual approach of ray tracing or modeling effects such as reflection and diffraction. Instead, it relies on building outlines and only considers line of sight between sender and receiver. This obstacles model has been integrated with the VEINS framework mentioned earlier, making the framework more suitable for VANET simulation.

2.4 Intelligent Vehicle Safety Applications

The National Highway Traffic Safety Administration (NHTSA), part of the United States Department of Transportation (DOT) identified and studied Intelligent Vehicle Safety Applications for VANETs in [4].Specific requirements were laid down per application. Table 2.2 lists a few of the applications with a short description.

Of these applications, the two applications of interest to this thesis are explained in detail below, and the requirements as listed in [4] are also mentioned. It is expected that vehicles participating in a VANET will be equipped with an On Board Unit (OBU), which may interface with the vehicle's central computer, other on board sensors and will be responsible for providing communication capability to the car. Before we talk about the applications, terms used in the application requirements are defined:

Transmission mode: Describes whether the transmission is triggered by an event or if it is sent periodically

Minimum Frequency: The rate at which a transmission should be repeated

Table 2.2: Public Safety Applications [4]

Application	Description
Approaching Emergency Vehicle Warning	Provides the driver a warning to yield the right of way to an approaching emergency vehicle.
Post-Crash Warning	This in-vehicle application warns approaching traffic of a disabled vehicle (disabled due to an accident or mechanical breakdown) that is stuck in or near traffic lanes, as determined using map information and GPS.
Traffic Signal Violation Warning	Warns the driver to stop at the legally prescribed location if the traffic signal indicates a stop and it is predicted that the driver will be in violation.
Left Turn Assistant	Provides information to drivers about oncoming traffic to help them make a left turn at a signalized intersection without a phasing left turn arrow.
Blind Merge Warning	Warns a vehicle if it is attempting to merge from a location with limited visibility (either for itself or for the oncoming traffic) and another vehicle is approaching and predicted to occupy merging space.
Pedestrian Crossing Information at Designated Intersections	Provides an alert to vehicles if there is danger of a collision with a pedestrian or a child that is on a designated crossing.
Curve Speed Warning	Aids the driver in negotiating curves at appropriate speeds
Work Zone Warning	Work zone safety warning refers to the detection of a vehicle in an active work zone area and the indication of a warning to its driver.
Cooperative Adaptive Cruise Control	Cooperative adaptive cruise control will use vehicle-to-vehicle communication to obtain lead vehicle dynamics and enhance the performance of current adaptive cruise control
Pre-Crash Sensing	Pre-crash sensing can be used to prepare for imminent, unavoidable collisions.

Allowable Latency: Maximum duration of time allowable between when information is available to be transmitted and when it is received.

Maximum Required Range of Communication: Maximum communication distance between two units that is needed to effectively support a particular application.

2.4.1 Approaching Emergency Vehicle Alert

This application allows public safety vehicles belonging to police, fire and emergency services in a state of emergency warn the vehicles that are approaching them or are in their path using the VANET. The drivers are then alerted if in case they need to yield to the emergency vehicle. The On Board Unit of the public safety vehicle in a state of emergency will send out broadcast messages which will alert the recipient vehicle's driver with relevant information.

Communication Requirements:

- Communication from vehicle to vehicle
- One-way communication
- Point-to-multipoint communication
- Transmission mode: event-driven
- Minimum frequency (update rate) : 1 Hz
- Allowable latency : 1 sec
- Data to be transmitted and/or received:
  - Emergency vehicle's position
  - Lane Information
  - Speed
  - Intended path/route
- Maximum required range of communication: 1000m

2.4.2 Post-Crash Warning

This application allows a disabled vehicle that has been involved in an accident or has experienced a sudden mechanical failure warn approaching vehicular traffic about its situation. On Board sensors of the vehicle help determine whether the vehicle was involved in a crash or is experiencing mechanical failure. The affected vehicle's OBU will send out broadcast messages to approaching vehicles and road side units alerting them of its state.

Communication Requirements:

- Communication from vehicle to road side unit or from vehicle to vehicle
- One-way communication
- Point-to-multipoint communication
- Transmission mode: event-driven
- Minimum frequency (update rate) : 1 Hz
- Allowable latency : 0.5 sec
- Data to be transmitted and/or received:
  - Position
  - Heading
  - Vehicle status
- Maximum required range of communication: 300m

Chapter 3

Research Work

The goal of this thesis was to implement and test a Public Safety application with two features : *Approaching Emergency Vehicle Alert* and *Post-Crash Warning* while satisfying the requirements laid down by the NHTSA. The developed application was to be deployed on a test bed capable of simulating all the aspects of VANETs, the application and also reflecting the result of the reaction of the vehicles due to the presence of the application,. The resulting reaction of a vehicle would be similar to how the driver of the vehicle would react to the information received, hence altering the flow of traffic in real time. The following subsections will explain in detail the work done towards achieving this goal.

3.1 Simulation Test Bed

Before implementing the application, it was first necessary to decide on a VANET simulation test bed and analyze the features and capabilities that it provided to develop, simulate and test the application. The simulator had to be capable of simulating the VANET as well as provide control to the application to mimic the behavior of a human driving the vehicle. Vehicles in Network Simulation (VEINS) was chosen as it provides a good platform with all the required features. The features provided and more details about the components that form VEINS will be explained in further detail in the following sections.

VEINS

VEINS is the outcome of the 'Veins Research Project' [17], and was developed by the team comprising of Christoph Sommer, Dr.Falko Dressler and David Eckhoff at the Computer Networks and Communication Systems Department, University of Erlangen, Germany. The main aim of this project was to realistically simulate Inter-Vehicle Communications (IVC). Also they realized the need of bidirectional coupling [20] between the traffic and network simulation needed to realistically simulate the effects of VANET applications on the mobility of the vehicles. To this end, a simulation framework was created by coupling a dedicated network simulator, OMNeT++ and a dedicated traffic simulator, SUMO (Simulation of Urban Mobility). TraCI (Traffic Control Interface) served as the glue between the two simulators making bidirectional coupling a possibility. This lets the network simulation also influence the traffic simulation and vice versa.

Following are some of the important features:

- Completely open source, hence can be extended as required
- Relies on a trusted vehicular mobility model.
- Models IEEE 802.11p and IEEE 1609.4 DSRC/WAVE network layers in full detail
  - Multi-channel operation
  - QoS channel access
  - Noise and interference effects
- Simulations can be deployed on a single workstation or extended to a computer cluster.
- Obstacle model which accounts for shadowing effects caused by buildings.

SUMO 0.13.1 (Simulation of Urban Mobility) [8]

SUMO is a fast and portable microscopic road traffic simulator developed by Institute of Transportation Systems at the German Aerospace Center using standard C++. It is open sourced, licensed under the GPL. It models the mobility of vehicles in a microscopic manner, meaning that every vehicle in the simulation is modelled explicitly and has its own route. Other characteristics of the vehicle such a acceleration, deceleration and length vary between vehicle types. The following are the features provided by SUMO:

- Open Source
- Time-discrete vehicle movement - where the position of each vehicle in the scenario is updated every time step.
- Multiple vehicle types like cars, trucks, vans, each with unique characteristics like acceleration, decelaration and top speed associate with them.
- Multi lane streets with lane changing
- Vehicles follow right of way rules
- OpenGL graphical user interface which provides a graphical view of the ongoing simulation with features like locating ang tracking vehicles during runtime.
- Interoperability with other applications during runtime
- Scenarios form OpenStreetMaps [13] can be imported. OpenStreetMaps is an online wiki of the world map, which allows users to update and maintain maps. Also, it allows exporting of map data in several formats. SUMO supports importing
  - Road networks
  - Speed limits
  - Lane counts
  - Traffic lights

  - Turn restrictions

- All configuration and data input like route and network definitions are accepted in the from of XML files.

Also, it provides a suite of other supporting applications which help import/generate the network and traffic for a simulation. The following applications are of interest to this thesis work:

Sumo-gui : Provides a Graphical User Interface to the simulation, enabling the user to observe the simulation in action. Allows display properties relating to the vehicles and road networks to be modified. Has a feature enabling the user to follow a particular vehicle also available. Displays information pertaining to the different components forming the scenario graphically.

Netconvert: Enables the user to import road networks from different formats and generate a road network that conforms with SUMO's format. This tool provides support to import maps from OpenStreetMaps [13].

Duarouter: Computes the shortest possible routes between source and destination pairs for a vehicle given the road network and other parameters.

Polyconvert:Enables the user to import other components like point of interests and buildings from OpenStreetMaps.

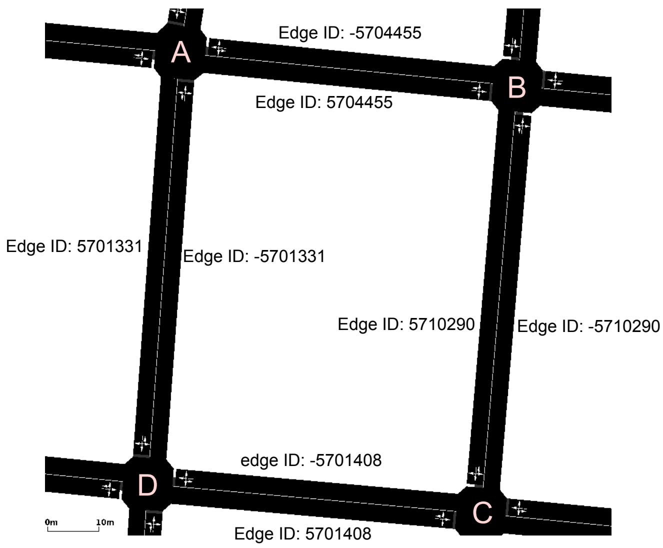

Figure 3.1: Road Network in SUMO

Figure 3.1 depicts a typical road network in SUMO. The figure has been explicitly labeled in order to explain it in detail in this section. An ’Edge’ in SUMO refers to the road connecting two junctions together. Every edge in the network has a unique identifier associated with it. The corresponding Edge IDs for every edge have been labeled in the figure. Also, the Edge IDs in the opposite direction only differ in their sign. Junctions in the figure are labelled A,B,C and D. Vehicle routes in SUMO are defined as a string which is a collection of Edge IDs. For example, a circular route starting from junction A and ending back at the same point would be defined as ”5704455 5710290 -5701408 -5701331”. A single edge may be split into multiple lanes. Here, every edge has a single lane.

OMNeT++ 4.2

OMNeT++ is a C++ based object oriented discrete event network simulation framework with a generic architecture and can be used to simulate the following:

- Wired and Wireless communication networks
- Protocol modeling
- Multiprocessors and distributed hardware systems

OMNet++ was developed at the Technical University of Budapest, Departmetnt of Telecommunications and is available for free for academic use. It provides the infrastructure and tools for writing simulations, and is not a ready to use Network Simulator by itself. A simulation model in OMNeT++ consists of a well architected arrangement of reusable ’modules’ which are connected to each other and can pass messages between each other. The smallest building block in a simulation model is called the ’simple module’. Simple modules are written in C++, and can be made to behave as desired. Simple modules are grouped together into compound modules. Both simple and compound modules communicate by sending messages to each other’s ’gates’ over connections that span between modules or other compound modules.

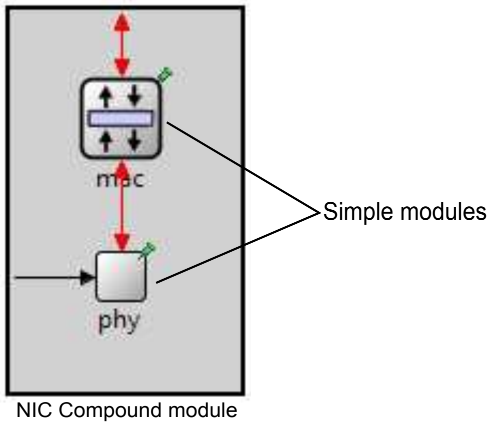

Figure 3.2: Compound module in OMNeT++

A compound module forming a Network Interface Card (NIC) in OMNeT++ is shown in Figure 3.2. Simple modules can be used to model algorithms using C++ wherever necessary.

OMNeT++ provides the tools required to define the structure of the entire system by providing the following features:

- Hierarchically nested modules.
- Inter-Module communication using messages through channels.
- Flexible module parameters.
- Network Description language (NED) used to define network topology.
- Graphical and text editing of NED files.

An OMNeT++ simulation model consists of the following:

- .ned files coded using the Network Description language (NED) which describe the position and connections beween various modules. Also, values for parameters relating to the simple modules can be defined here.
- .msg files which contain the message definitions defining various message types with data fields which are later translated by OMNeT++ to C++ classes.

- C++ source files for simple modules.
- Simulation kernel used for managing the simulation and simulation class library, all in C++.
- .ini file is used to specify modifiable parameters explicitly for all the modules involved at any level of the hierarchy.

MiXiM (Mixed Simulator)

MiXiM is an OMNeT++ modeling framework for simulating fixed and mobile wireless networks. MiXiM stands for 'Mixed Simulator'. It provides detailed models of various layers of the OSI model along with accurate models for radio wave propagation and interference estimation. It also has a vast library of networking protocols that can be incorporated into any simulation. Since it follows the module hierarchy of simple modules forming more complex compound modules, a great deal of flexibility is available when it comes to developing new compound modules. Also, user defiend simple modules can be added to the framework, enabling the user to model custom compound modules. This feature is necessary for the implementation of this thesis work where the application is implemented as a simple module and then incorporated into a more complex compound module defining the vehicle in the simulation. It supports simulation of networks with up to 1000 nodes.

The VEINS simulation framework comes as part of MiXiM, adding support for IEEE 802.11p, IEEE 1609.4 and IEEE 1609.3 (WSMP). Also, the obstacles model discussed in section 2.3.3 is available as a simple module.

TraCI [11]

TraCI stands for Traffic Control Interface, and this is what gives the outside world access to an already running simulation in SUMO in real time. It uses a TCP based client/server based architecture to provide access to SUMO. As part of the VEINS project, the TraCI Scenario Manager module in OMNeT++ was introduced as part of the MiXiM framework. This gives us the ability to couple OMNET and SUMO together as shown in Figure 2.5. It enables a network simulation running in OMNeT++ to send TraCI commands which control the traffic simulation in SUMO. The TraCI protocol explains in detail the messaging and message format to be used. There are different classes of TraCI commands, those of interest to this thesis are listed below.

Vehicle Value Retrieval: This suite of commands queries SUMO about the value of a certain variable pertaining to a particular vehicle from the last time step.

Route Value Retrieval: This suite of commands queries SUMO about information pertaining to a certain vehicle route in the simulation.

Change Vehicle State: This suite of commands sets the value of variables related to a particular vehicle.

3.2 Use Case - Application Design

This section contains the system level use case developed with an aim to satisfy the requirements and also describe the behavior of the entire system in all possible simulation scenarios. All non Public Safety vehicles will be referred to as ’vehicles’, and a Public Safety vehicles (Police/Ambulance/-Fire) will be referred to as ’PS-Vehicle’. Only PS-Vehicles are capable of transmitting an Emergency Wave Short Message (WSM). All vehicles are capable of transmitting an Accident WSM.

3.2.1 Primary Scenario

1. The vehicle/PS-Vehicle is not involved in an accident, nor has it heard about any emergency from an approaching PS-Vehicle. Each alternate scenario discussed below branches off of the primary scenario, and hence is numbered to indicate the same.

3.2.2 Alternate Scenarios

PS-Vehicle Declares a State of Emergency

2. The operator of the PS-Vehicle decides to declare an emergency, and uses the provided in-vehicle interface activating the sirens and flashing lights. This is indicated in the SUMO-GUI application by the PS-Vehicle's color alternating between red and blue.
3. The PS-Vehicle will start broadcasting an Emergency-WSM every 1000 ms on the Control Channel with priority 1 (highest priority). The WSM will contain the following information:

- Current Position of the vehicle - Obtained using the 'position' TraCI command.
- Current speed - Obtained using the 'speed' TraCI command.
- Intended route - Obtained using the 'route id' TraCI command.
- Emergency Number - A number unique to this instance of emergency, user-defined in the simulation's .ini file in OMNET++.
- Emergency Sequence Number - A number unique to this WSM

4. The PS-Vehicle will travel at an increased speed till it reaches its destination.

Vehicle/PS-Vehicle is involved in an accident

2. The vehicle/PS-vehicle is involved in a collision. The collision and the severity of it is detected and assessed by the on board sensors (ABS, Airbag, Accelerometer, Gyroscope).

3. The vehicle/PS-Vehicle starts broadcasting an ACCIDENT-WSM every 1000 ms with priority 1 (Highest Priority) on the Control Channel.The WSM will contain the following information:

- Current Position of the vehicle - obtained using the 'position' TraCI command.

- Route - obtained using the 'route id' TraCI command.
- Vehicle status - A number ranging from 0 through 10, 10 being most critical - user-defined in the simulation's .ini file in OMNET++.

Vehicle Receives an Emergency- WSM

2. A vehicle in the simulation receives an Emergency WSM, the source could be a PS-Vehicle or another intermediate vehicle which is forwarding the broadcast packet.
3. When the distance between the vehicle and the PS vehicle is greater than 1000m, or the message has expired (time elapsed from creation to reception of the message $>$1sec), the message is discarded and will not be broadcasted any further.
4. When the distance between the vehicle receiving the message and PS-Vehicle is less than 1000m, the following is done:

- If the distance is greater than 300m, the message is retransmitted only if it is a new message and was never received before. This check can be accomplished using the Emergency Number/Emergency Sequence Number (ESN) duple where the emergency number is unique to that instance of emergency. The ESN is unique to a WSM sent out by the public safety vehicle and is incremented for every WSM sent out.
- The PS-vehicle's intended path, current position obtained from the message and the vehicle's route and current position are used to determine if the vehicle needs to give way to the PS-Vehicle.
- If yes, and the distance from the PS-Vehicle is in the range of 0 to 250m, the vehicle is removed from the simulation in SUMO, hence giving way to the PS-Vehicle. As soon as the PS-Vehicle passes by and starts moving away, the vehicle is reintroduced to the simulation in SUMO right where it was removed.

Vehicle Receives an Accident-WSM

The vehicle (A) receives an ACCIDENT-WSM, from a vehicle involved in a collision (B).
If the distance between the collision affected vehicle (B) and the recipient vehicle (A) is greater than 300m, or if the message has expired (time elapsed $>$1sec), it is discarded.
If the distance is less than 300m, the following is done:

- A's current position, intended path, and the collision affected vehicle-B's position and intended path are compared to determine if A is approaching B or not.
- If B is in A's path and is in the range of 50 to 0m, the car comes to a complete stop and changes color to red in SUMO.

PS-Vehicle Receives an Emergency- WSM

2. PS-Vehicle-'A' receives an Emergency -WSM about a PS-Vehicle-'B' in a state of emergency.
3. When PS-Vehicle A is not in a state of emergency, it behaves similar to any other vehicle in the scenario as described earlier.
4. When PS-Vehicle A is also in a state of emergency, its behavior is described in the exception scenario later in this document.

PS-Vehicle receives an Accident-WSM

Everything in 2.4 applies to this scenario except, the PS vehicle will take up the responsibility to report the accident to the control/dispatch center. This is indicated in the simulation by a change in color of the PS-Vehicle.

3.2.3 Exception Scenario

PS-Vehicle in a state of emergency receives an Emergency WSM from another PS-Vehicle.

3. PS-Vehicle-'A' receives an Emergency - WSM about a PS-Vehicle-'B'. Both A and B are in a state of emergency.

4. When the distance between the vehicle and the PS vehicle is greater than 1000m, or the message has expired (time elapsed from creation to reception of the message >1sec), the message is discarded and will not be broadcasted any further.
5. When the distance is less than 1000m and greater than 500m, 'A' retransmits the message once after 100ms regardless of whether it is in B's path or not.
6. A PS-vehicle in a state of emergency will not give way to another PS-vehicle in a state of emergency.

3.3 Implementation

The public safety application was implemented as a simple module in OMNeT++ to be a part of MiXiM. The entire application was written using C++ and the NED language. The application provides the following parameters as mentioned in Table 3.1 which can be modified on a per vehicle basis using the OMNeT++ simulator's .ini file associated with that simulation.

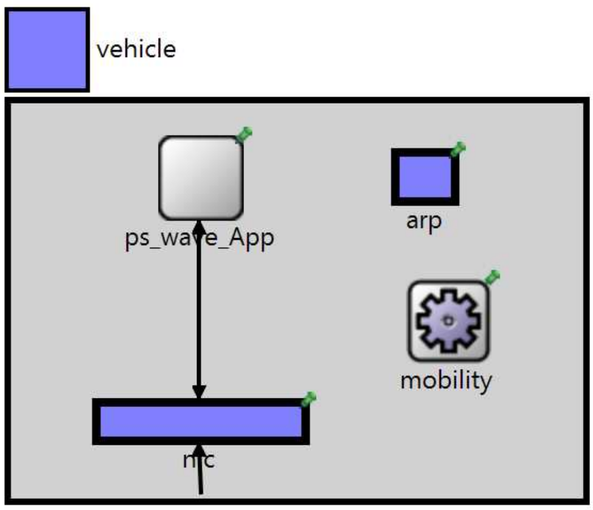

Figure 3.3: Vehicle compound module in OMNeT++

In OMNeT++, a vehicle was defined as a compound module as shown in figure 3.3, composed of the following sub-modules:

Table 3.1: Application Parameters

Parameter	Type	Default Value	Description
isPS_Vehicle	bool	FALSE	If true, this vehicle will behave as a Public Safety Vehicle Capable of declaring emergency.
declareEmergencyAfter	double	0 sec	Time in seconds after which the PS-Vehicle should declare a state of emergency
emerReTransInt	double	1 sec	Time interval in seconds between transmission of consecutive Emergency WSMs
emergencyNumber	integer	50	An Emergency Number that should be unique to this instance of emergency for a particular simulation run.
isAccidentVehicle	bool	FALSE	If true, this vehicle will be involved in an accident in the simulation.
causeAccidentAfter	double	5 sec	Time in seconds after which this vehicle should simulate being involved in an accident.
accidentNumber	double	45	An Accident Number unique to this instance of accident for a particular simulation run.
accidentSeverity	double	5	Indicates the severity of the accident on a scale of 1 to 10, 10 being most critical.
accidentReTransInt	double	1 sec	Time interval in seconds between transmission of consecutive Accident WSMs

- *PS WAVE App* - The simple module that was implemnted to model the public safety application adhereing to the system level use case defined in section 3.2.
- *IEEE 802.11p NIC* - Compound module consisting of MAC 1609.4, MAC 802.11p and PHY 802.11p
- *TraCI Mobility* - Mobility module for hosts controlled by the TraCI Scenario Manager
- *Arp* - Module responsible for address resolution.

The IEEE 802.11p NIC is a compound module as shown in figure 3.4, composed of the following sub-modules:

- MAC IEEE 1609.4
- MAC IEEE 802.11p

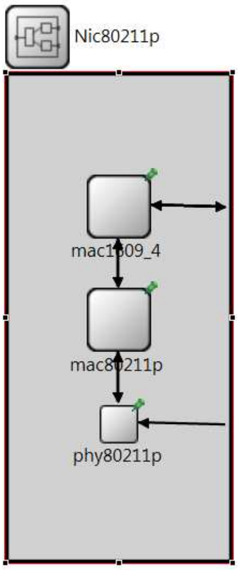

Figure 3.4: IEEE802.11p compound module in OMNeT++

- PHY IEEE 802.11p

Support for the following TraCI commands was added in order to make it feasible to develop the application: ***Vehicle Value Retrieval***: This suite of commands queries SUMO about the value of a certain variable pertaining to a particular vehicle from the last time step.

- *road id*: Returns the id of the edge the vehicle was on.
- *lane id*: Returns the lane id the vehicle was on.
- *route id*: returns the id of the route the vehicle is currently following
- *edges*: Returns the ids of the edges the vehicle's route is made up of
- *lane position*: the position of the vehicle along the lane.

Route Value Retrieval: This suite of commands queries SUMO about the vaules pertaining to a certain defiend route in the simulation.

- *edges*: Returns the ids of the edges that the route covers.

Change Vehicle State: This suite of commands sets the value of variables related to a particular vehicle.

- *color*: Sets the vehicle's color.
- *move to*: Moves the vehicle to a new position based on the lane ID and the lane position passed.
- *add*: Adds a new vehicle to the simulation with the passed parameters.
- *remove*: Removes the desired vehicle form the traffic simulation.

Chapter 4

Simulation Setup

The developed application was tested to ensure that it satisfied the NHTSA requirements and functioned as expected according to the system level use case. The simulation setup here consisted of first modeling the mobility of the vehicles using SUMO as desired and then configuring the application in OMNeT++ prior to running the simulation. The following sections will explain how this was achieved.

4.1 SUMO Simulation Setup

This section describes the steps involved in preparing a SUMO traffic simulation to be used for simulating the VANETs.

4.1.1 Exporting the map from OpenStreetMap

The road map on which the VANET is to be simulated is imported from openstreetmap.org by manually selecting the desired area and using the 'export as .osm' option to download a map.osm file, which is an xml file with an openstreetmap defined schema.

4.1.2 Editing the map using JOSM:

Java Open Street Map Editor is a tool used to manually edit .osm files. The imported map.osm also contains other information like railway networks, power distribution lines, rivers, points of interest, walkways, bicycle routes and other features apart from the road network which are of no interest to us. Java Open Street Map Editor is used to edit the map. The unnecessary nodes are deleted and if required, new nodes are added to create new routes and also to create buildings in the scenario. When a node is deleted using JOSM, it is not actually removed from the .osm file, but instead an ’action’ attribute with a value ’deleted’ is added to that node. All such nodes are removed by using an open source command line xml toolkit called XML Starlet using the following commands.

```
./xml ed -d "/osm/node[@action='delete']"
<input_map.osm> output.osm

./xml ed -d "/osm/way[@action='delete']"
<output.osm> map.osm
```

The resulting .osm file contains only the road network and buildings that are required for the simulation scenario.

4.1.3 Generating a Sumo network using NETCONVERT

SUMO provides a command line application called Netconvert which is used to import and generate road networks that can be used by SUMO and other tools present in the SUMO package. Also, it projects the geographic co-ordinates (Latitude and Longitude) to x-y coordinates and also applies the required offset to translate the axes to the first quadrant. The following command line arguments are passed to achieve the desired road network

```
./netconvert --osm-files map.osm
                    -o map.net.xml
                    --junctions.join
                    --remove-edges.isolated
                    --remove-edges.by-vclass "rail_slow
                     , rail_fast , bicycle , pedestrian"
                      -v
```

Table 4.1 describes the options used in further detail.

Table 4.1: Description of options used in NETCONVERT

Option	Description
–osm-files	.osm File Name
-o	output .net.xml File Name
–junctions-join	joins junctions that are close to each other
–remove-edges.isolated	removes isolated edges
–remove-edges.by-vclass	removed the edges related to the mentioned vehicle class
-v	verbose output of the application

4.1.4 Generating vehicle routes using DUAROUTER

The resulting map.net.xml file from the previous step is now used to generate a Trips file using the RandomTrips python script provided as part of the SUMO package. This script generated a set of random trip definitions- demand for routes between two points in the road network.These are then used in the creation of vehicle routes. The following command line command is used

```
./randomTrips.py -n map.net.xml -o map.trips.xml
-b 0 -e 500 -v
```

Table 4.2 describes the options used in further detail.

Table 4.2: Description of options used in RandomTrips

Option	Description
”-n”	.net.xml file to be used
”-o”	output .trips.xml File Name
”-b”	start time in seconds, 0 in this case
”-e”	end time in seconds, 500 sconds in this case
”-v”	verbose description of the application

The map.trips.xml along with the map.net.xml is now used to generate the vehicle routes using the Duarouter.exe application which again is a part of the SUMO package. The following command is used

```
./duarouter.exe -n map.net.xml -t map.trips.xml
```

```
-o map.rou.xml --remove-loops -b 0 -e 500\\
-v -ignore-errors
```

Table 4.3 described the options used in further detail.

Table 4.3: Description of options used in DUAROUTER

Option	Description
”-n”	.net.xml file to be used
”-t”	.trips.xml file to be used
”-o”	output routes file name
”–remove-loops”	removes routes that have loops within them
”-b 0”	start time in seconds, 0 in this case
”-e 500”	end time in seconds, 500 sconds in this case
”-v”	verbose description of the application
”–ignore-errors”	discards instances where a valid route cannot be generated

4.1.5 Adding obstacles to the simulation

The Obstacles module in OMNeT++ that is responsible for modeling obstacles requires an xml file with a special schema definition used to define the obstacles in the scenario.
Polyconver.exe, an application part of the SUMO package is used to extract the polygons/buildings from the .osm file using the following command.

```
./polyconvert.exe --osm-files map.osm -n map.net.xml
-o map.poly.xml
```

The obstacles.xml file shares a similar schema to the .poly.xml file and can be directly used, but only after inverting Y-axis. This is required as the co-ordinate axes in sumo are as that in the first quadrant, whereas the ones in OMNeT are that of the fourth quadrant. This inversion is achieved using a simple MATLAB script.

4.1.6 SUMO Configuration File

Each scenario in sumo has a .sumo.cfg file associated with it which points to the corresponding .net.xml and .rou.xml files to be used, and also the start and end time of the simulation. The xml schema used is as described below:

```
<configuration>
        <input>
                <net-file value="rural_sm.net.xml"/>
                <route-files value="rural_sm.rou.xml"/>
        </input>

        <time>
                <begin value="0"/>
                <end value="500"/>
        </time>
</configuration>
```

4.1.7 SUMO-GUI Configuration

To display the right vehicle colors as set by the Public Safety Application in OMNET through TraCI, the 'View Settings' in the SUMO-GUI application are set to display the 'given/assigned vehicle color'.

4.2 OMNeT++ Simulation Setup

The simulation in OMNeT++ can be controlled by supplying the required modifiable parameters per module that is being used in the simulation using the .ini file corresponding to that simulation. The parameters used, and their specific values are discussed below with excripts form the .ini file.
The playground size is set to the maximum size of the map in order to prevent vehicles from going out of bounds. The mobility of vehicle in x and y direction is handled by the mobility model and hence set to zero. But the position on the z-axis is set to a constant value representing the height of the antenna from the ground.

```
*.playgroundSizeX = 5000m
```

```
*.playgroundSizeY = 5000m
*.playgroundSizeZ = 50m
*.world.useTorus = false
*.world.use2D = false

*.host[*].mobility.x = 0
*.host[*].mobility.y = 0
*.host[*].mobility.z = 1.895
```

The following are the parameters set for the TraCI Scenario Manager, which is responsible for the loopback TCP connection between OMNeT++ and SUMO.

```
*.manager.updateInterval = 1s
*.manager.host = "localhost"
*.manager.port = 9999
*.manager.moduleType = "org.ps_application.vehicle"
*.manager.autoShutdown = false
*.manager.margin = 25
*.manager.launchConfig = xmldoc("map.launchd.xml")
```

The following are the IEEE 802.11p Network Interface Card specific parameters:

```
*.connectionManager.pMax = 20mW
*.connectionManager.sat = -94dBm
*.connectionManager.alpha = 2.0
*.connectionManager.carrierFrequency = 5.890e9 Hz
*.connectionManager.sendDirect = true

*.**.nic.mac1609_4.serviceChannel = 2
*.**.nic.mac80211p.txPower = 20mW
*.**.nic.mac80211p.bitrate = 18Mbps
*.**.nic.phy80211p.sensitivity = -94dBm
*.**.nic.phy80211p.maxTXPower = 20mW
```

```
*.**.nic.phy80211p.useThermalNoise = true
*.**.nic.phy80211p.thermalNoise = -110dBm
*.**.nic.phy80211p.decider = xmldoc("config.xml")
*.**.nic.phy80211p.analogueModels = xmldoc("config.xml")
*.**.nic.phy80211p.usePropagationDelay = true
```

The following are used to enable the use of obstacles in the scenario:

```
*.annotations.draw = true
*.obstacles.debug = true
*.obstacles.obstacles = xmldoc("map.obstacles.xml")
```

The following are the parameters provided by the Public Safety Application module, set on a per vehicle basis.

```
[Config Public Safety Vehicle]
*.host[*].ps_wave_App.isHighway = false
*.host[*].ps_wave_App.reTransmitEmerWsm = true
*.host[*].ps_wave_App.reTransmitDistMin = 100
*.host[*].ps_wave_App.reTransmitDistMax = 900
*.host[0].ps_wave_App.isPS_Vehicle = true
*.host[0].ps_wave_App.declareEmerAfter = 10s
*.host[0].ps_wave_App.emerTransFreq =1s

[Config Accident]
*.host[1].ps_wave_App.isAccidentVehicle = true
*.host[1].ps_wave_App.causeAccidentAfter =10s
*.host[1].ps_wave_App.accidentNumber=667
*.host[1].ps_wave_App.accidentSeverity = 7
*.host[1].ps_wave_App.accidentTransFreq = 1s
```

4.3 Simulation Scenarios:

The procedure mentioned in the previous sections was used to generate the following scenarios that were used to test the application.

4.3.1 Grid/Downtown

This scenario is made up of a 2000m X 2000m grid with blocks of sides 250m. This grid was created using JOSM and buildings were added to each block which are modeled as obstacles in OMNeT++. The obstacles introduced in the network simulation help model the shadowing and multipath fading effects that the communicating nodes will encounter in a typical typical downtown scenario. All roads in the map have 2 lanes in each direction. Figure 4.1 depicts the road network in SUMO.

4.3.2 Rural

This map was imported from OpenStreetMap by manually selecting a section of rural area of 10,900 square meters. This map is a typical example of a rural area where the roads are spaced far apart, with mostly flat topography, with almost no obstacles. All roads in the map have a single lane in each direction. Figure 4.2 depicts the road network in SUMO.

4.3.3 Highway

This map too was imported from OpenStreetMap, and is an actual 6.7 stretch of Interstate I-83. The speed limit per edge was set to 31.29 m/s which equals 70 Miles/hr. This scenario is an example of the usual highways where opposite lanes are separated by the use of a median. Figure 4.3 depicts a section of the highway road network in SUMO.

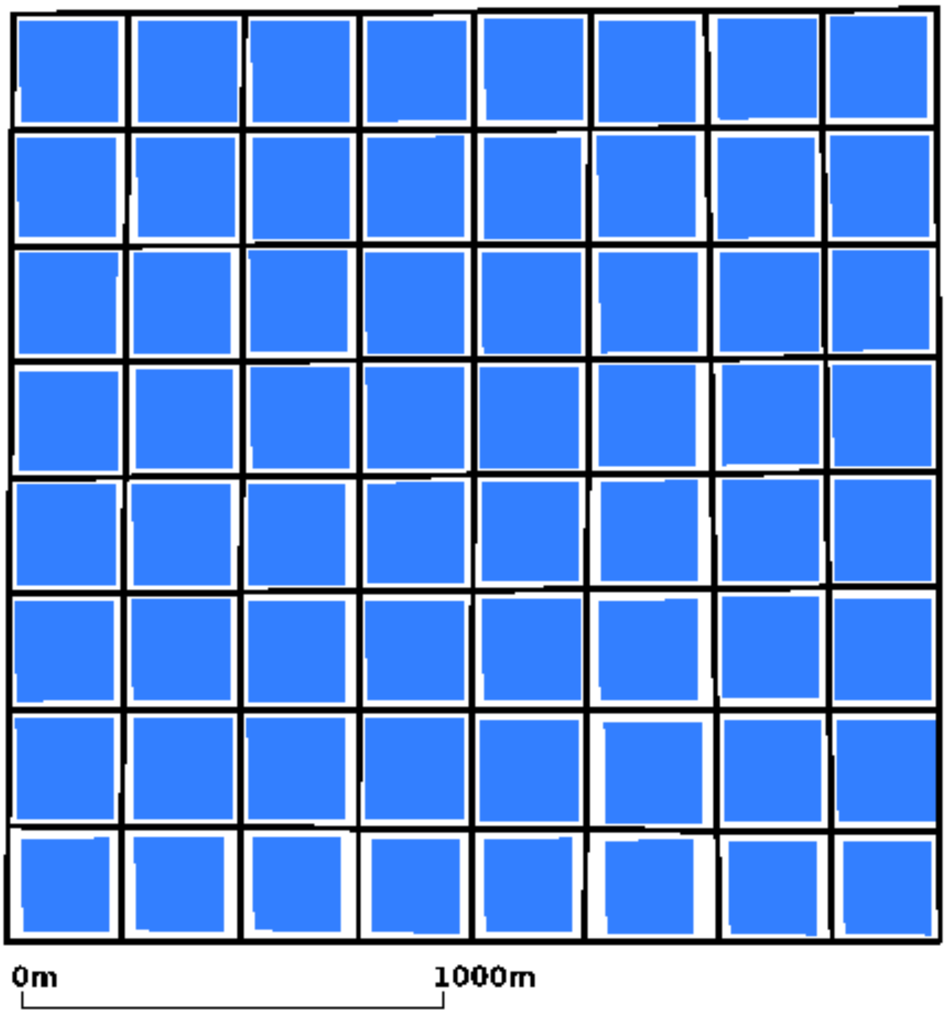

Figure 4.1: Grid-Downtown Road Network in SUMO

Figure 4.2: Rural Road Network in SUMO

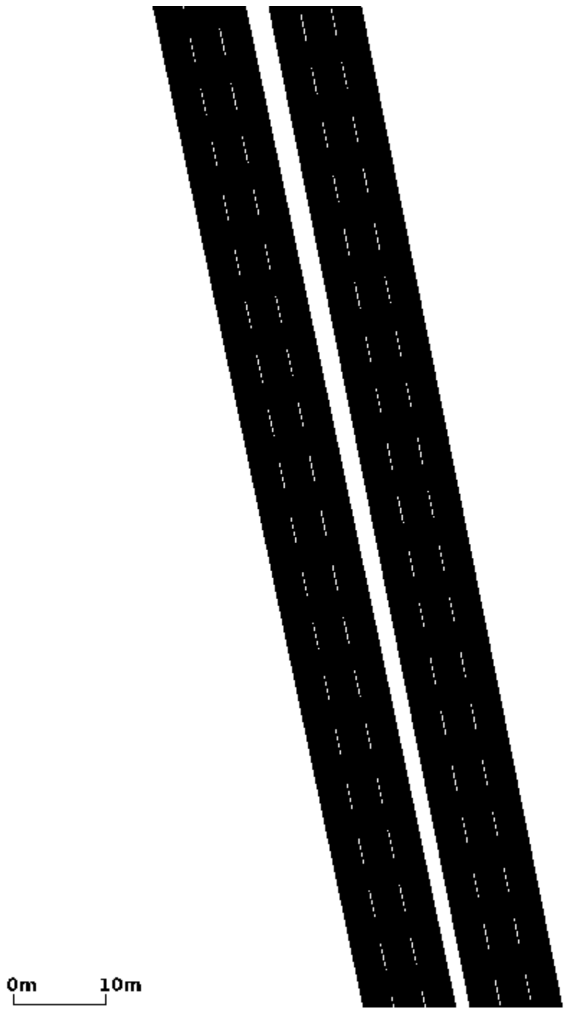

Figure 4.3: Highway Road Network in SUMO

Chapter 5

Simulation Results And Analysis

This chapter presents an in depth discussion of how the simulation scenarios presented in Chapter 4 were used to simulate VANETs running the Public Safety application. The results collected are presented and analyzed in the following sections.

5.1 Result Collection Criteria

Before collecting results, it was critical to first understand and decide on what data was needed to gauge the performance and working of the applications features. The following was tested in the simulation scenarios.

5.1.1 Allowable Latency

The NHTSA defines this as the maximum duration of time allowable between when information is available to be transmitted and when it is received. This translates to the time elapsed from when the WSM was built to be transmitted in either the Public Safety Application or the Vehicle involved in an accident to when that WSM is received by another vehicle present within the maximum required range of communication. In order to collect this data, every WSM generated was time-stamped with the simulation time at that moment, and then was deducted from the simulation time at reception of the message by the application at the receiving vehicle and the values were recorded as vectors.

According to the requirements mentioned in Chapter 2, the allowable latency for the Post Crash Warning feature is 0.5 sec and that for the Approaching Emergency Vehicle Alert is 1 sec.

5.1.2 Maximum Required Range of Communication

The NHTSA defines this as the maximum communication distance between two vehicles that is needed to effectively support a particular application. This could also be called the application range, the radius from the originating vehicle till where the application is active. In order to collect this data, the position of the vehicle generating the WSM at that simulation time was transmitted as part of the WSM and at the recipient vehicle, when the WSM was received, the co-ordinates of the two vehicles were passed to SUMO and the distance between them was recorded. In multi-hop situations, the WSM contained the co-ordinates of the vehicle where the WSM was first generated, which is either the Public Safety vehicle or the vehicle involved in an accident and not that of the intermediate vehicle which retransmitted the message, hence giving us true application range.
According to the requirements mentioned in Chapter 2, the Maximum Required Range of Communication for the Post Crash Warning feature is 1000m and that for Post Crash Warning is 300m.

5.1.3 Application's Adherence to the System Use Case

In all the test scenarios, the different situations in the Use Case were recreated and the behavior of the application was observed. The different situations were created by using predetermined routes for vehicles and causing Public Safety vehicles declare emergency or vehicles cause accidents at the required simulation time.

5.2 Simulation Scenarios

Unless explicitly specified, Approaching Emergency Vehicle Alert related WSMs were multi-hop as according to the use case requirements. Post Crash Warning related WSMs were all single hop, meaning they were not retransmitted by intermediate vehicles.

The rate of retransmission of the WSMs was 1 per second for both features as according to the NHTSA requirement.

5.2.1 Grid-Downtown

This scenario mimics the typical downtown scenario with buildings and other structures hindering the communication between nodes. The aim here was to assess the performance of the application in the worst possible case that VANETs can encounter. To achieve this, the route shown in Figure 5.1 was chosen. The public safety vehicle traveled from the south west corner while the other vehicles travelled in the opposite direction. This path has the maximum number of obstacles between the public safety vehicle and the approaching vehicles.
The approaching emergency vehicle feature of the application uses multi-hop to relay information in order to achieve a better communication range and get close to the requirement's specification in such scenarios. The multi-hop feature is further tested in order to determine the cases when it is effective and when it fails. The simulations performed and the results observed are discussed below:

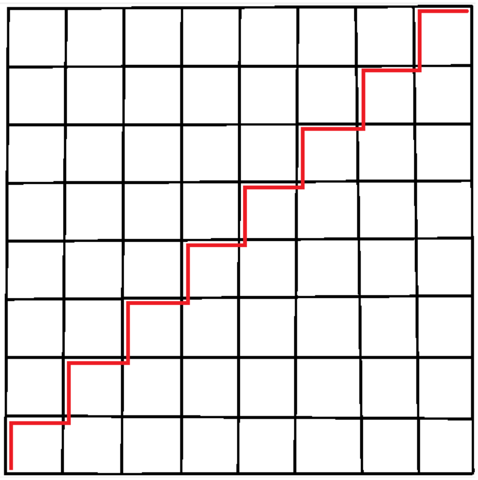

Figure 5.1: Grid: Vehicle route for worst case scenario

Multi-hop - Maximum allowable distance between nodes

This test was performed to assess the maximum distance between two vehicles in a downtown scenario before multi-hop breaks down. Two vehicles, one following the other were made to travel on the route indicated in figure 5.1 from top right to bottom left and the public safety vehicle was made to travel from bottom left to the top right along the same route, approaching the vehicles. The distance between the two vehicles was controlled by varying the time at which they were introduced in the scenario. Figure 5.2 plots the distance to the public safety vehicle. The separation between the two vehicles host 0 and host 1 (not the public safety vehicle) was 400m. Here, it was observed that no multi-hop message was being received by host 1. The distance between the them was then reduced to 300m, and Figure 5.3 plots the resulting distance to the public safety vehicle. Here it was observed that at simulation time of 194sec, host 1 received a multi-hop message from host 0. In the case of a perfect multi-hop example, one would expect to see the first message retransmitted by host 0 to be received by host 1. This is not always the case here as the relative position of the two hosts around an obstacle plays a major role in the successful error free reception of the message. Hence, it was observed that multi-hop fails when the two vehicles are more than 300 meters apart in the grid scenario.

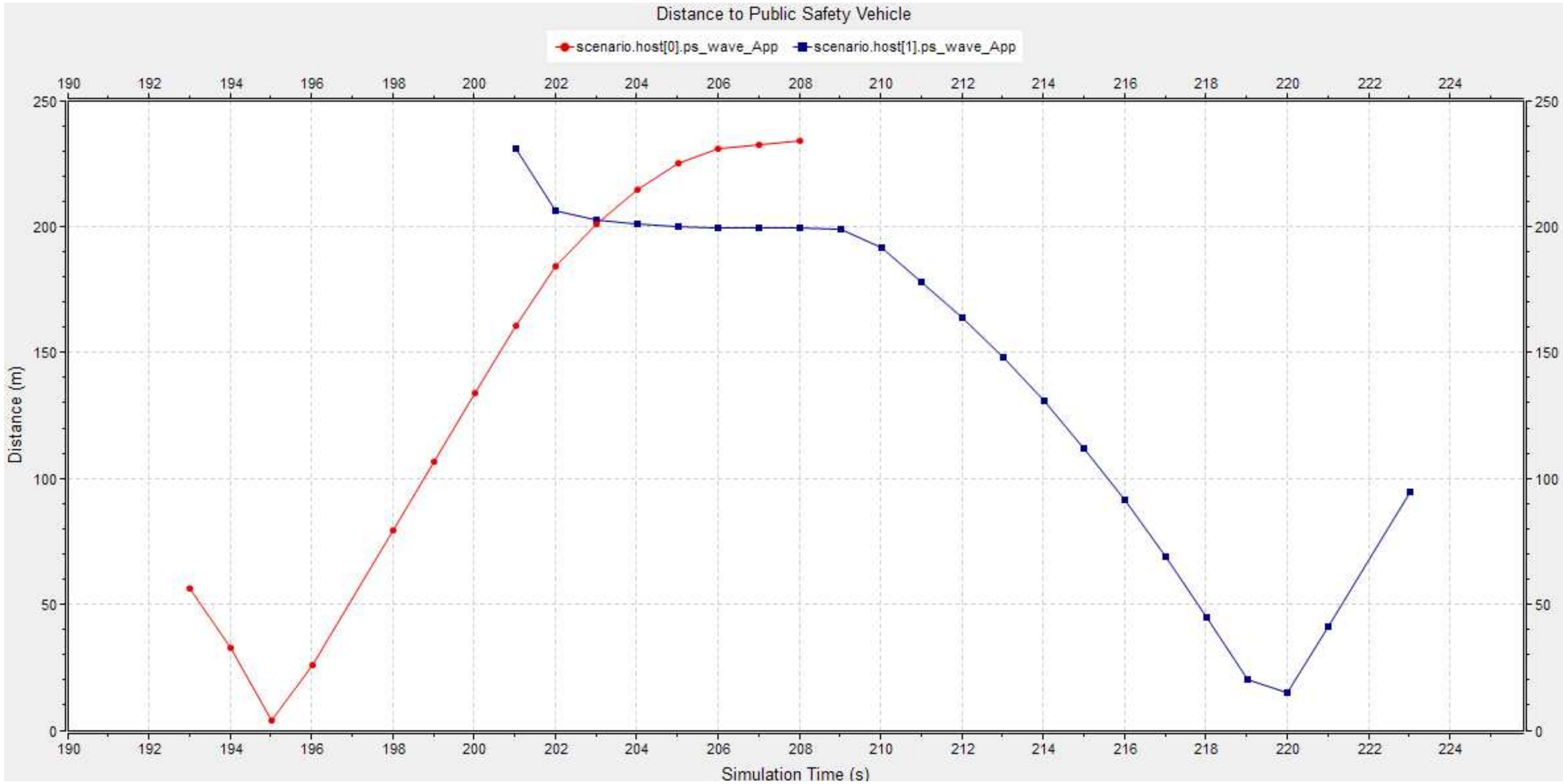

Figure 5.2: Multi-hop: 400m Distance between nodes

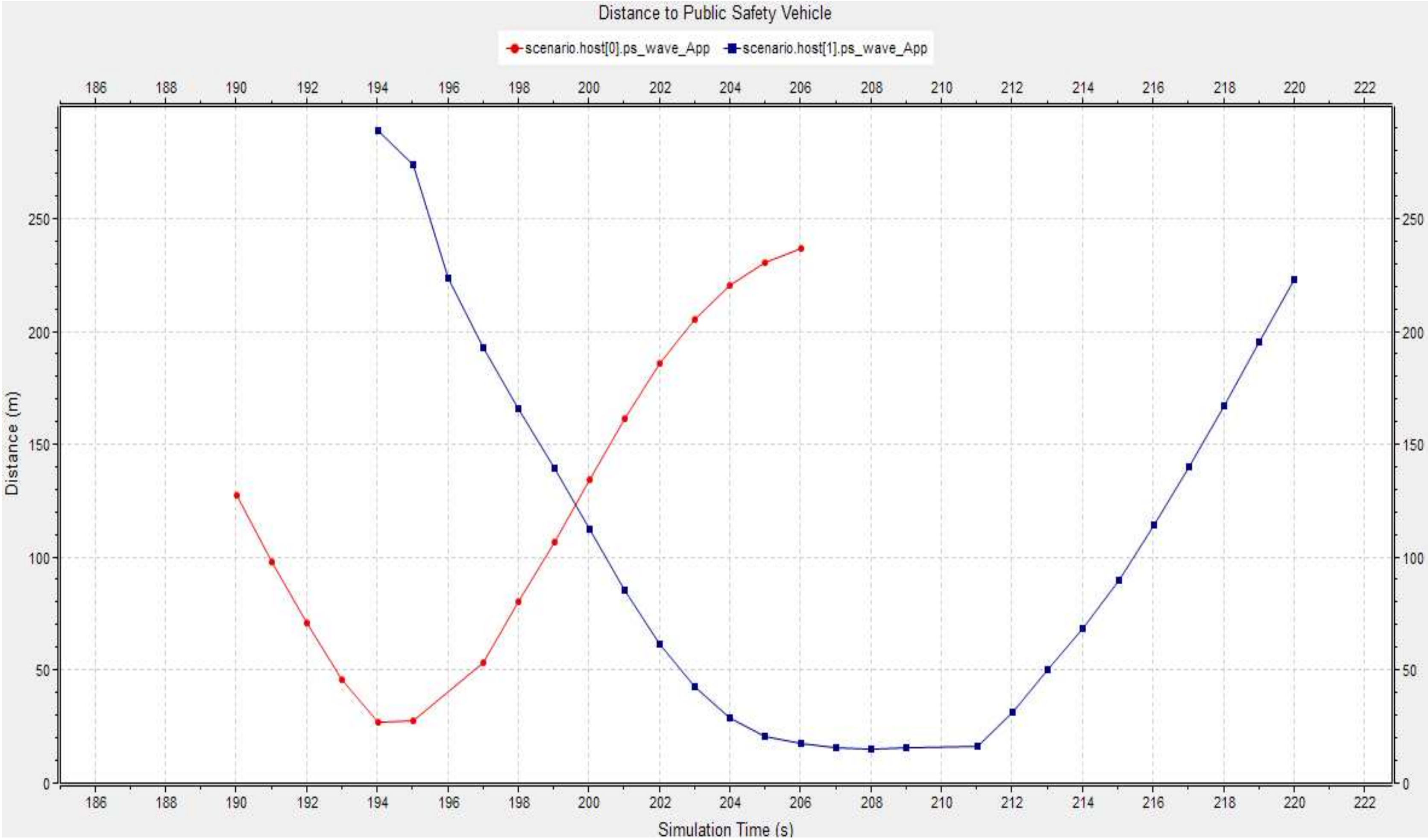

Figure 5.3: Multi-hop: 300m Distance between nodes

Multi-hop: Two groups of vehicles separated by a distance

This test was performed to observe the behavior of the multi-hop feature between two groups of vehicles separated by a considerable distance. To achieve this, two groups consisting of 4 vehicles each were made to travel from the top right corner to the bottom left corner along the route shown in Figure 5.1 while the public safety vehicle approached them in the opposite direction on the same path. Hosts 0, 1, 2 and 3 formed the first group whereas 4,5,6, and 7 formed the second group. The vehicles in each group travelled close to each other, biut a fixed distance could not be maintained between them as the distance varied slightly due to them being microscopically simulated by SUMO. Figure 5.4 plots the distance between the vehicles and the public safety vehicle when the two groups were separated by a distance of 300m. Here, it was observed that multi-hop works as all the vehicles in the second group receive the WSM at a simulation time of 139sec. Figure 5.5 plots the case where the two groups are separated by 600m, enough to make sure that multi-hop fails, and it is observed that it does in fact fail, with hosts 4,5,6, and 7 receiving the WSM only when the public safety vehicle is close to them. Hence, for multi-hop to work effectively in a downtown scenario, it is necessary to have intermediate vehicles or RSUs to relay the information across.

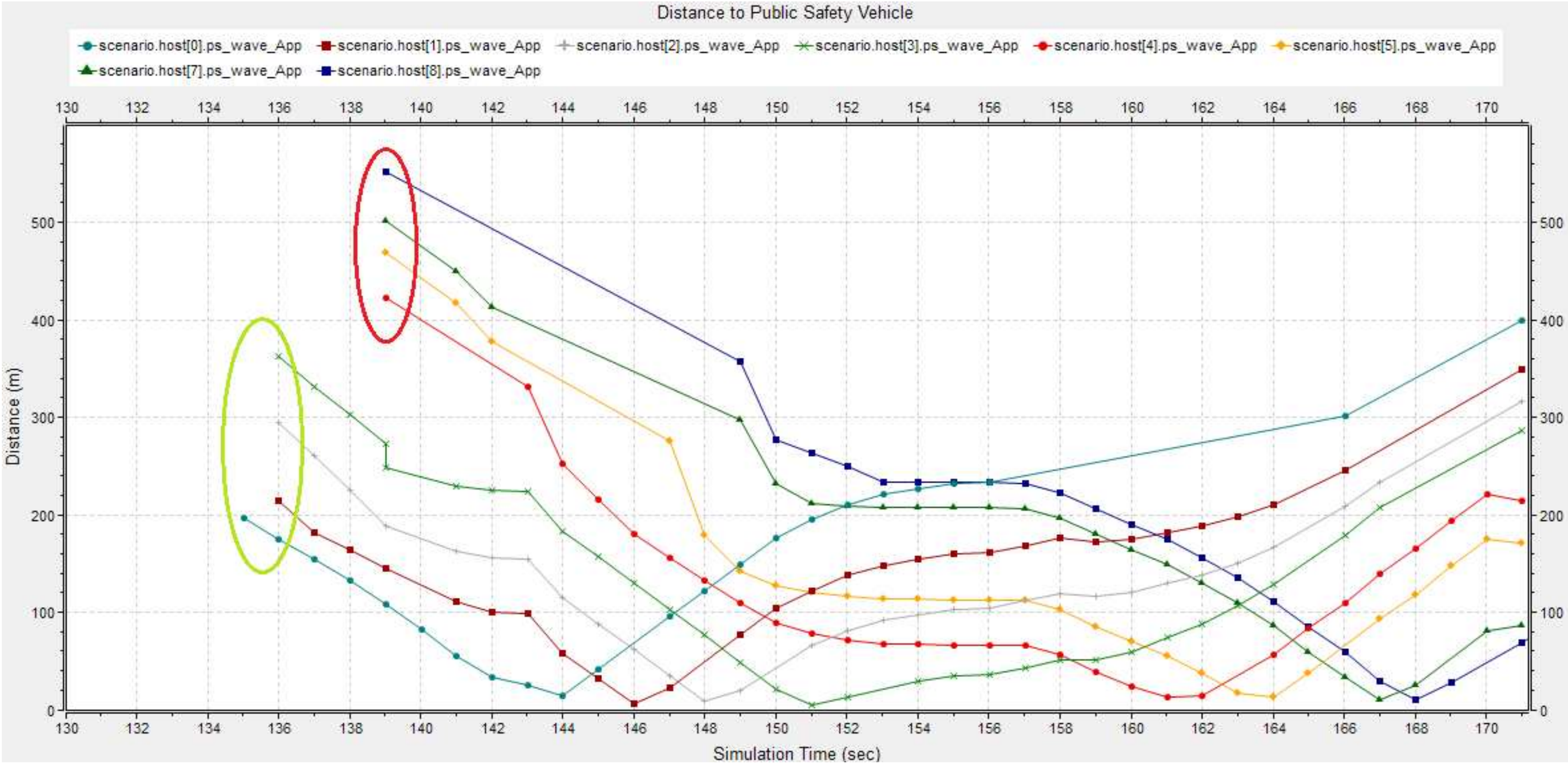

Figure 5.4: Multi-hop: 300m Distance between groups

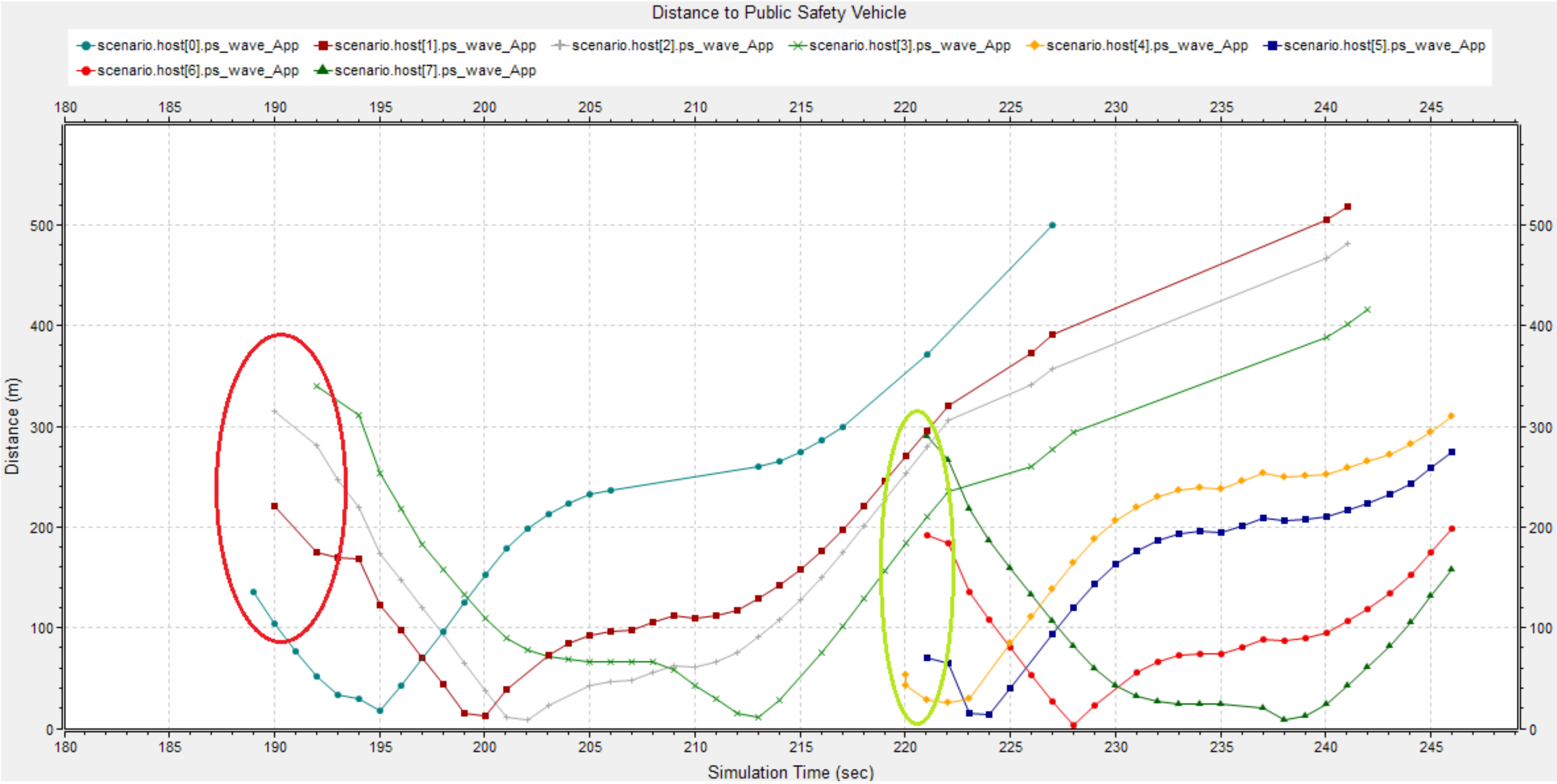

Figure 5.5: Multi-hop: 600m Distance between groups

No Multi-hop versus Multi-hop

This test was performed to test the effectiveness of multi-hop in increasing the application's range given there are intermediate vehicles available to relay the messages across. Eight vehicles were made to traverse the route shown in Figure 5.1 from the top right corner to the bottom left corner, and the public safety application approached them in the opposite direction. Figure 5.6 plots the distance to the public safety when multi-hop is disabled. Figure 5.7 plots the distance to the public safety vehicle when multi-hop is enabled. It was observed in Figure 5.6 that without multi-hop the application range was solely dependent on the public safety vehicle's vicinity to the other vehicles. In Figure 5.7, it was observed that with multi-hop enabled, host 8 (dark blue) received the first WSM at a simulation time of 131sec as compared to 150sec without multi-hop. The application range increased by 652m. It was observed that the WSM was passed on from one host to another as expected in a multi-hop scenario.

Figure 5.8 plots the latency per WSM received by the vehicle in the multi-hop scenario, and it was observed that the mean of the maximum observed latency per vehicle was 0.018sec, which is considerably lower compared to the 1sec requirement.

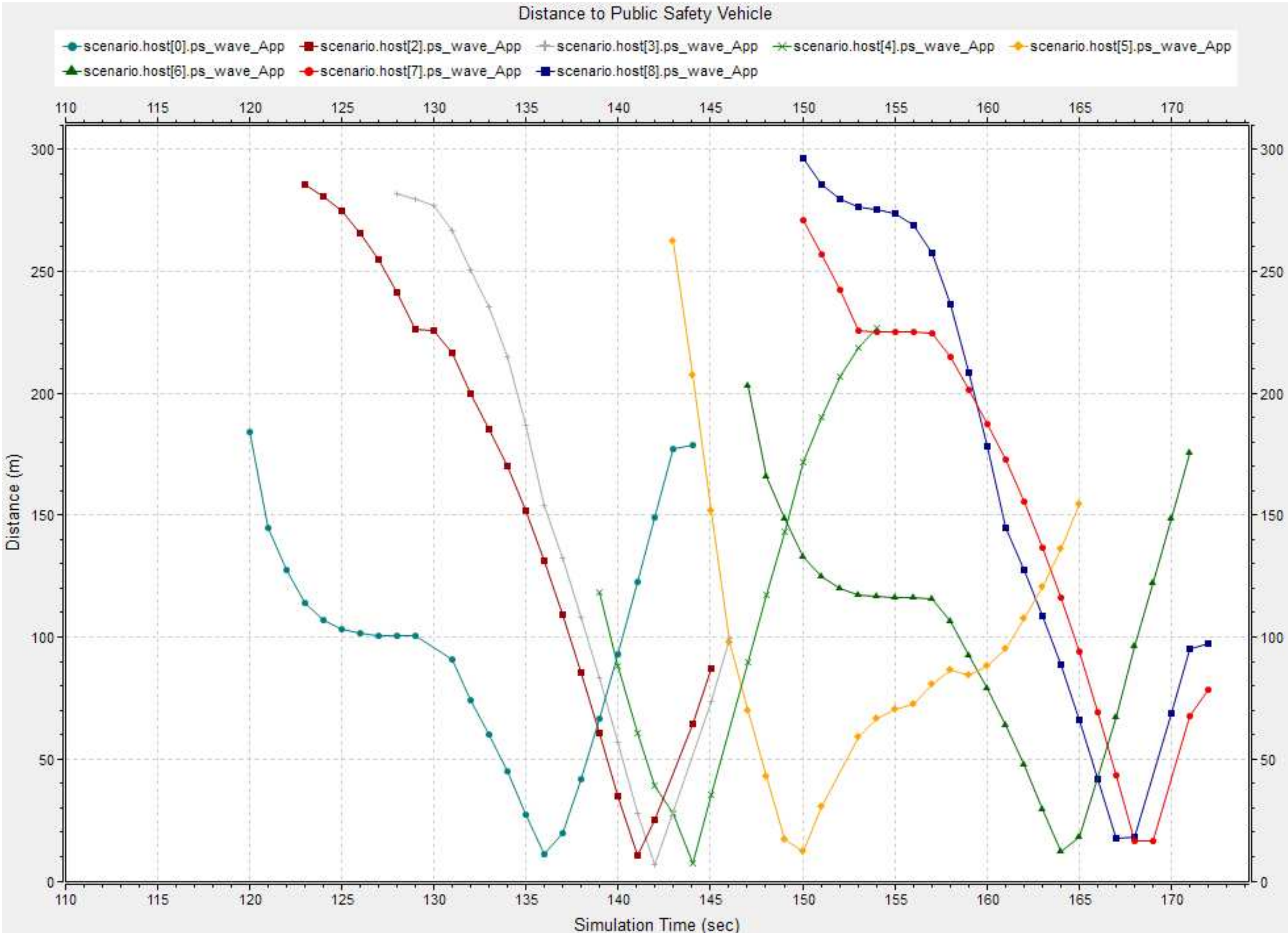

Figure 5.6: Grid: Distance to Public Safety Vehicle - No Multihop

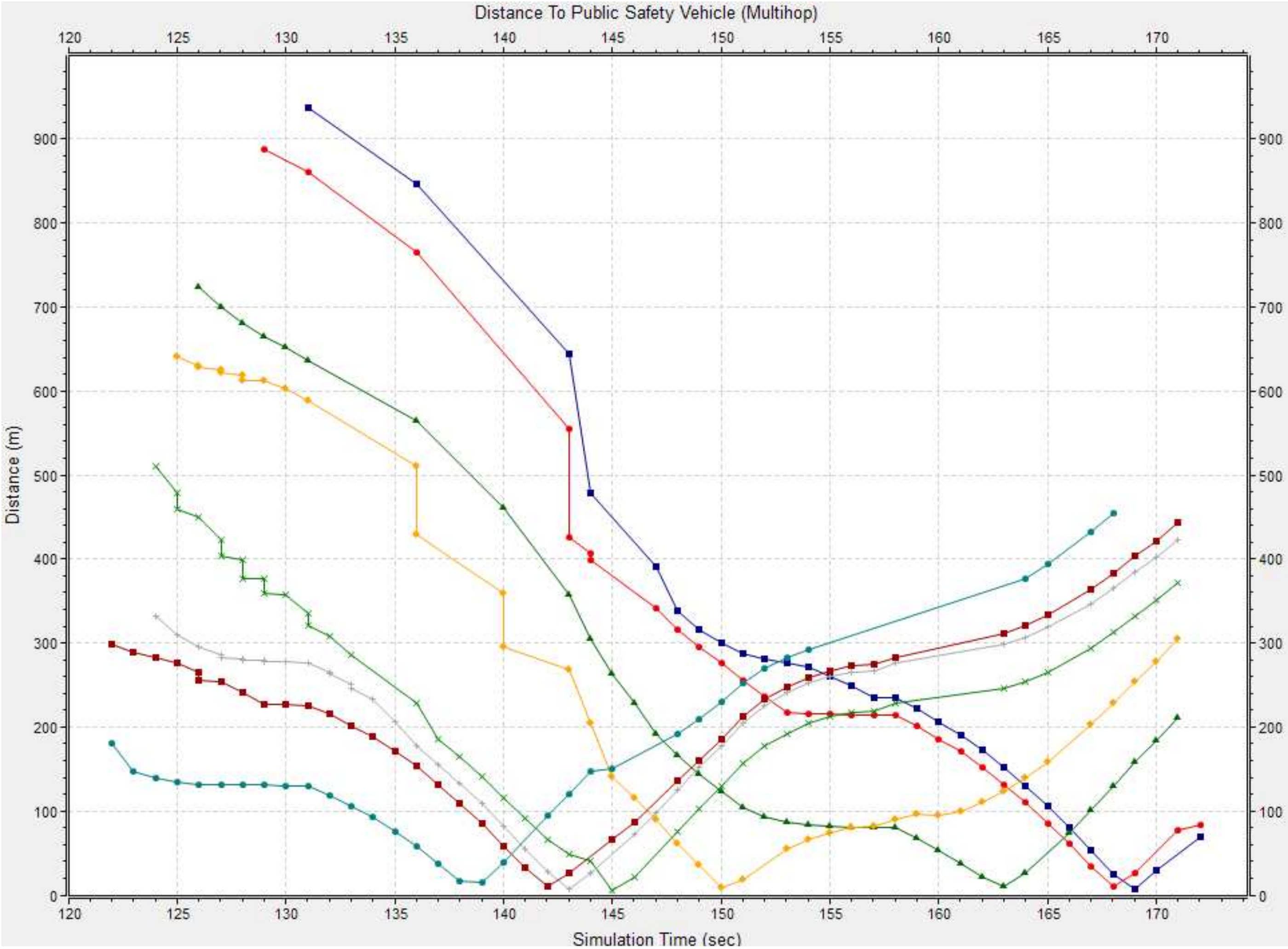

Figure 5.7: Grid: Distance to Public Safety Vehicle

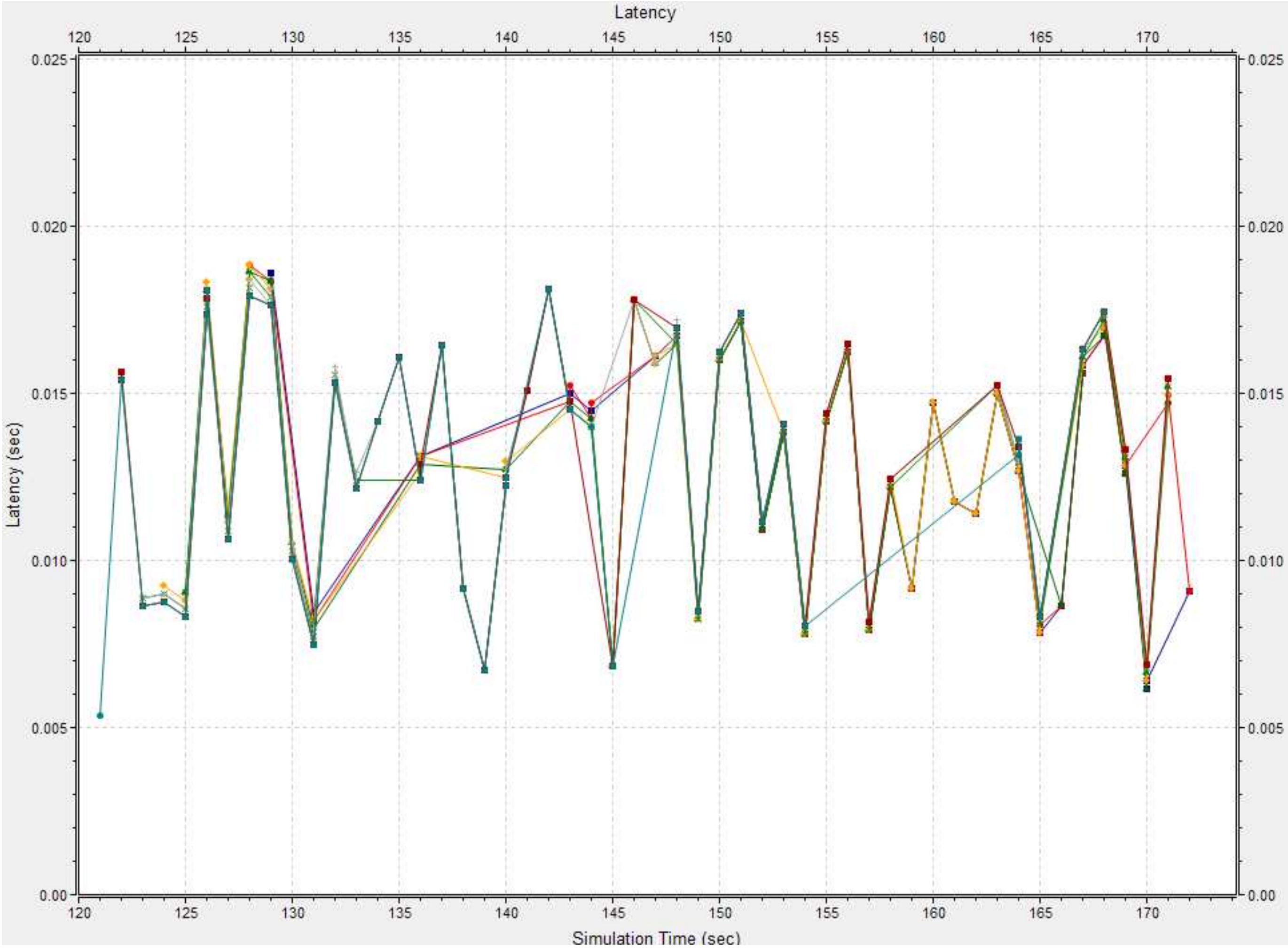

Figure 5.8: Grid: Latency - Approaching Emergency Vehicle Feature

Post-Crash Warning

For this scenario, the vehicles follow the same route as that in the no multi-hop versus multi-hop scenario and approach the accident that has occurred in the opposite lane. No multi-hop was implemented in case of the Post-Crash Warning feature as the application range requirement is just 300m. Figure 5.9 plots the distance to the crashed vehicle for every WSM received. It was observed that the vehicles started receiving the WSMs when at a distance of 295m from the crashed vehicle and then slowed down and came to a stop at a safe distance from the crashed vehicle and stayed there. This behavior was responsible for the unique pattern observed in the plot. Also, it was noted that the distance at which the vehicles came to a complete stop varies as they stop behind each other, hence increasing their distance from the crashed vehicle.

Figure 5.10 plots the latency per WSM received by the vehicle , and it was observed that the mean of the maximum observed latency per vehicle was 0.017sec, considerably lower than the 0.5sec requirement.

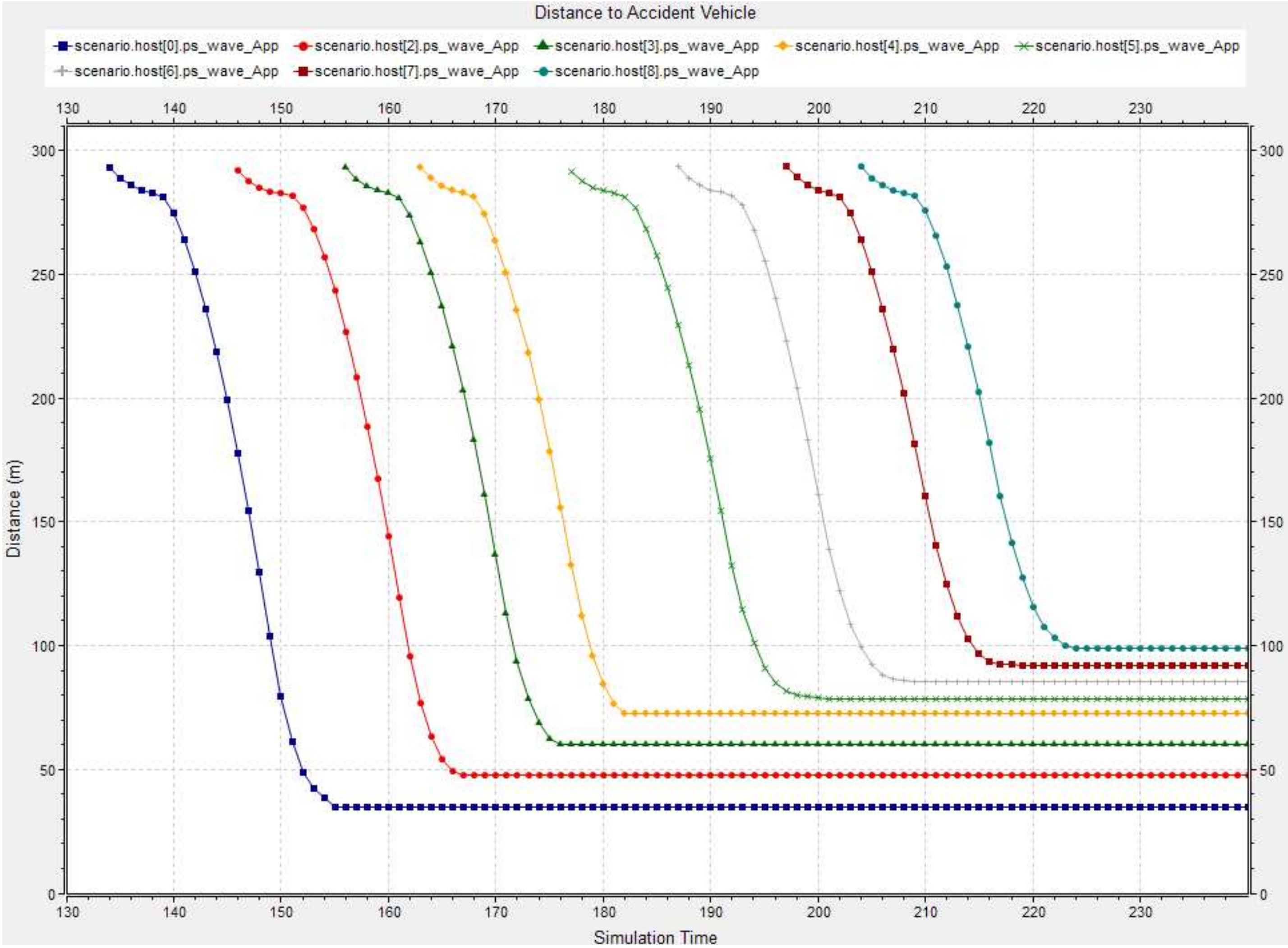

Figure 5.9: Grid: Distance to Accident Vehicle

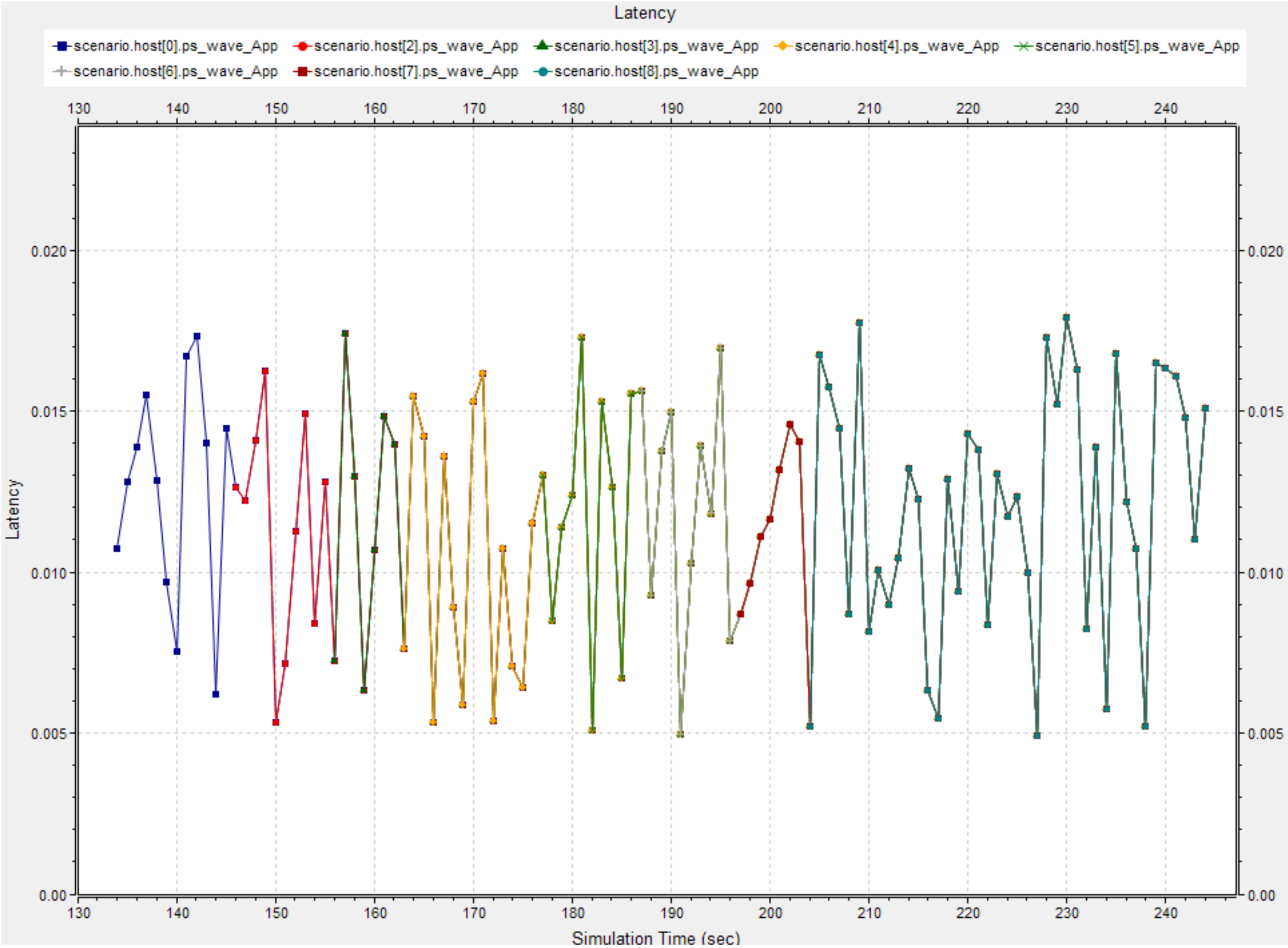

Figure 5.10: Grid: Latency - Post-Crash Warning Feature

Table 5.1 summarizes the results for the Grid-Downtown scenario.

Table 5.1: Grid: Summary of Results

Scenario	Mean of Max Communication Distance (m)	Mean of Maximum Latency (s)
Approaching Emergency Vehicle Alert - no multihop	237.75	0.017
Approaching Emergency Vehicle Alert - multihop	627.22	0.018
Post-Crash Warning	292.83	0.017

5.2.2 Rural

This scenario models the rural setting of a road network. No obstacles are present here, and the traffic is less dense at a maximum speed of 45mph.

Approaching Emergency Vehicle Alert

Eight vehicles were present in the scenario, along with a public safety vehicle in a state of emergency. Also, multi-hop was enabled for this scenario as we would definitely want to take advantage of it whenever possible to increase the application's range. Figure 5.11 plots the distance to the public safety vehicle for every WSM received. A few varied patterns were observed in this case. Host 3 and host 4 were never directly in the path of the public safety vehicle, and hence their graph does not take a significant dip as the other vehicle's graph does. But these vehicles played an important role of acting as an intermediate hop for host 5 which received the WSM well in advance as a result of the multi-hop feature. Host 5 did not receive a single WSM from simulation time 28sec to 48sec as it was not in communication range of any of the other vehicles. Hosts 1, 2, and 6 also received the WSM well in advance. The mean of maximum communication distance was 1057.83m, and surpassed the 1000m requirement.

Figure 5.12 plots the latency per WSM received by the vehicle , and it was observed that the mean of the maximum latency per vehicle was 0.017sec, considerably lower than the 1sec requirement.

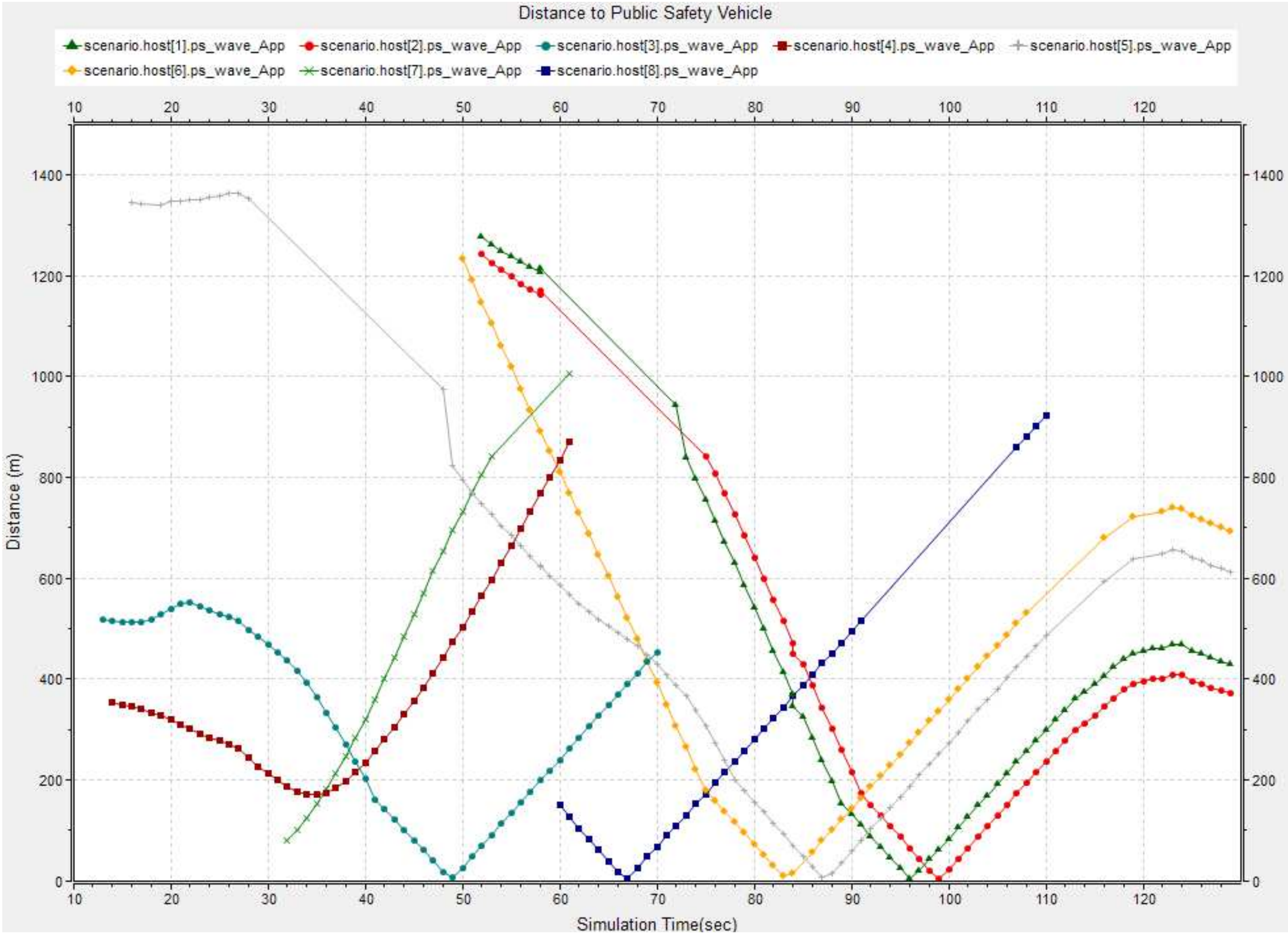

Figure 5.11: Rural: Distance to Public Safety Vehicle

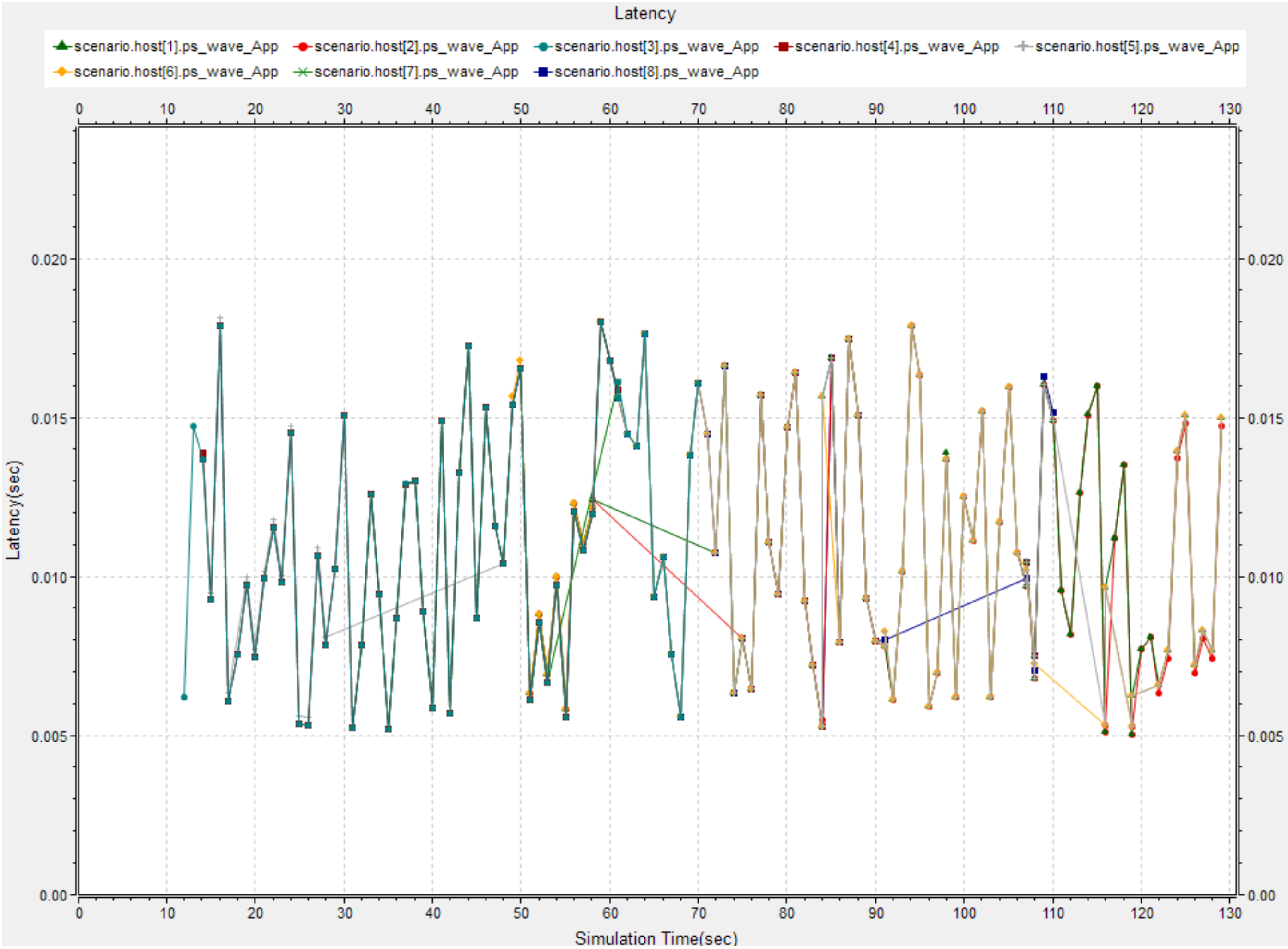

Figure 5.12: Rural: Latency - Approaching Emergency Vehicle Feature

Post Crash Warning

Eight vehicles were present in the scenario and each of them approached the crashed vehicle using the same route by following each other. Figure 5.13 plots the distance to the crashed vehicle for every WSM received. The specific pattern observed in the plot is due to the vehicles approaching the crashed vehicle, slowing down and then stopping at a safe distance away from it as they still receive the WSMs.

Figure 5.14 plots the latency per WSM received by the vehicle , and it was observed that the mean of the maximum latency per vehicle was 0.018sec, considerably lower than the 0.5sec requirement.

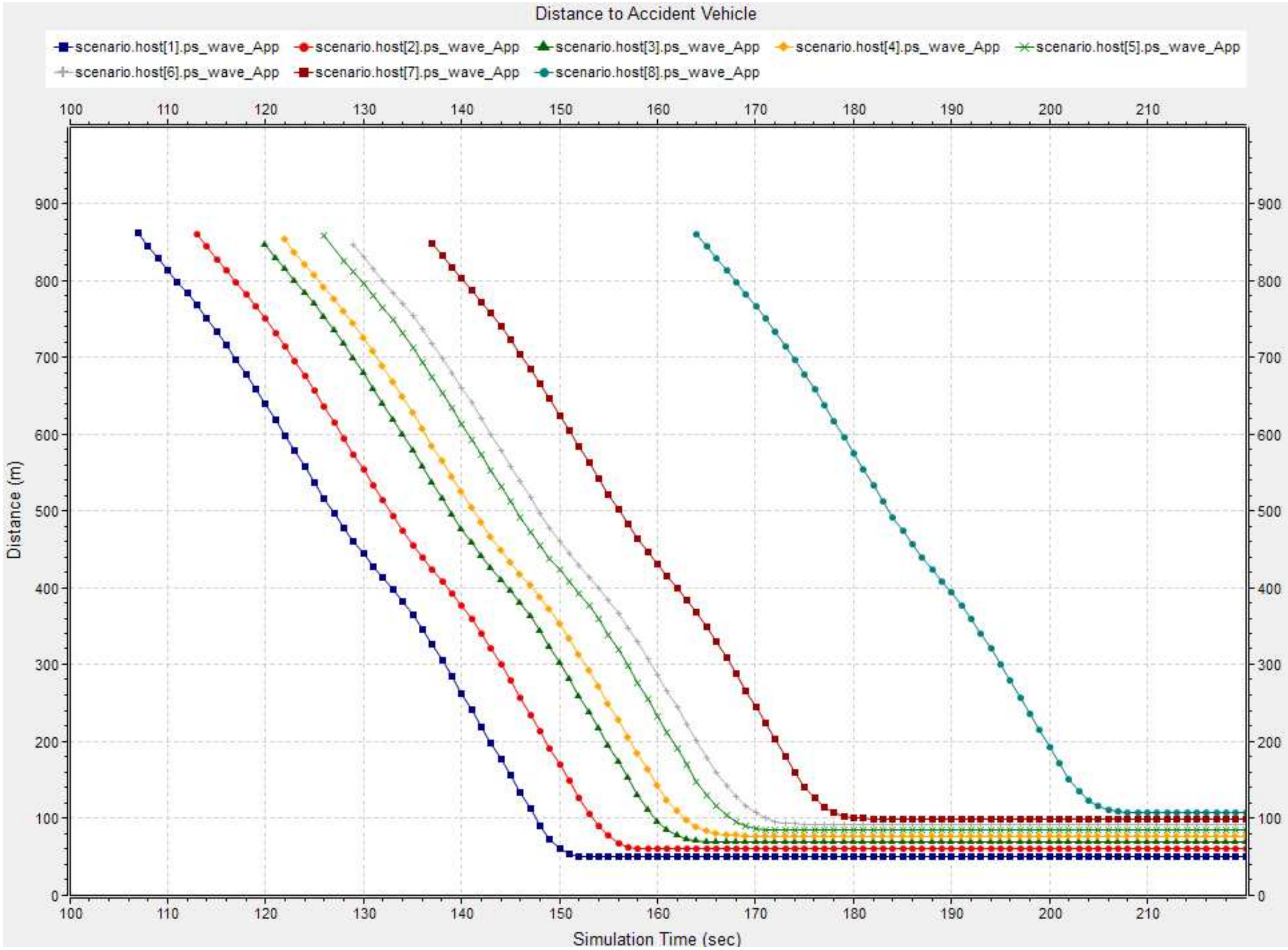

Figure 5.13: Rural: Distance to Accident Vehicle

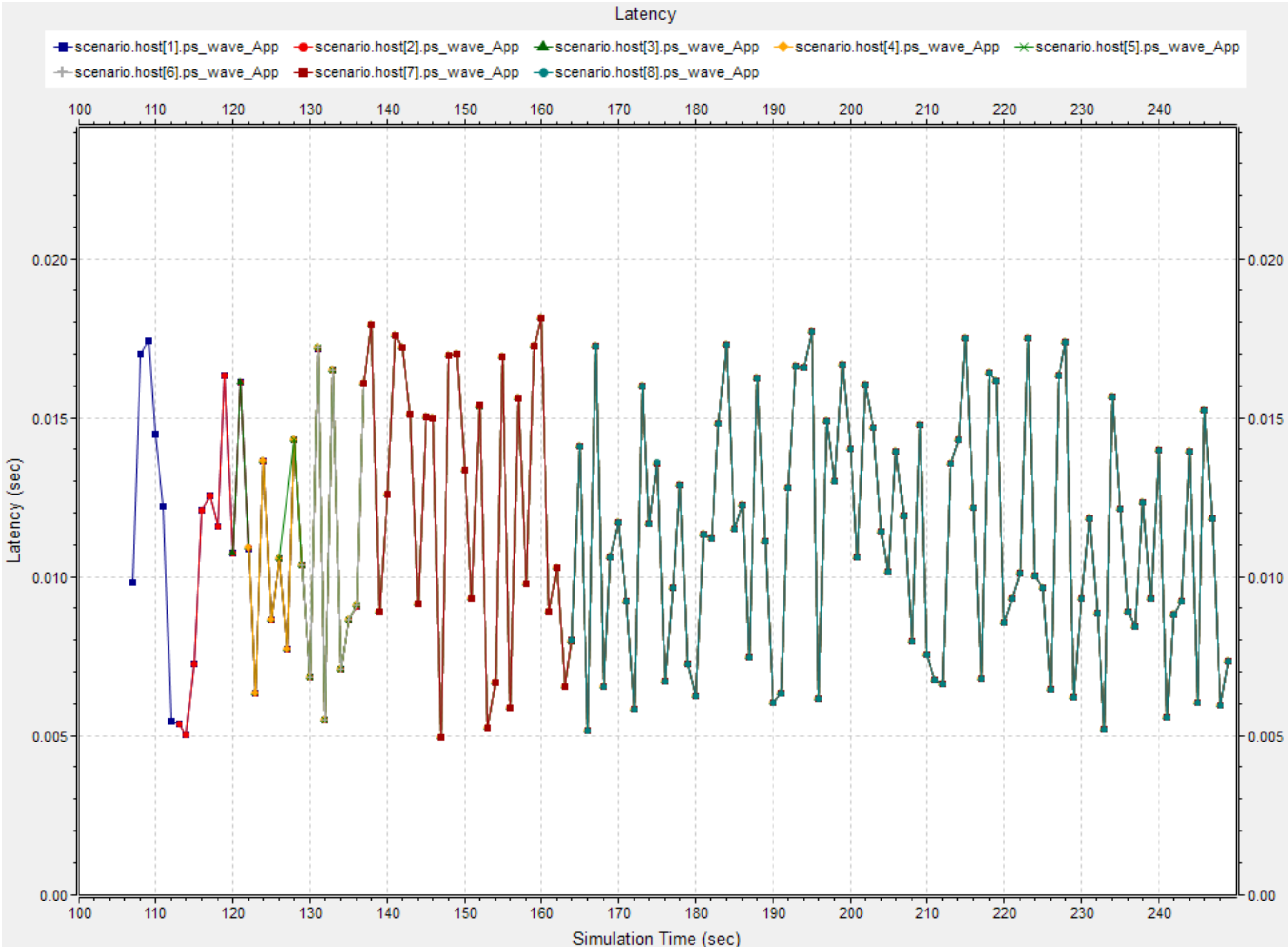

Figure 5.14: Rural: Latency - Post-Crash Warning Feature

Table 5.2 summarizes the results for the Grid-Downtown scenario.

Table 5.2: Rural: Summary of Results

Scenario	Mean of Max Communication Distance (m)	Mean of Maximum Latency (s)
Approaching Emergency Vehicle Alert	1057.83	0.017
Post-Crash Warning	854.09	0.018

5.2.3 Highway

This scenario models the highway setting of a road network. No obstacles are present here and vehicles travel a high speed of 75mph.

Approaching Emergency Vehicle Alert

Eight vehicles were made to approach the public safety vehicle in a state of emergency in the opposite direction. . Figure 5.15 plots the distance to the public safety vehicle for every WSM received. Due to the presence of multi-hop, it can be seen that the vehicles that are behind a vehicle that is closer to the public safety vehicle and has already received a WSM receive the same WSM well in advance, and the application range increases with every additional hop taken by the WSM. The same occurs as the public safety vehicle moves away from the group of vehicles. Also, because this is a highway scenario, the NED parameter for highway was set to true, causing the vehicles in the opposite lane not to give way. Thus, though the curves dip, its only because they get close to the public safety vehicle that is in the opposite side of the highway. They do not give way to the public safety vehicle and just continue on their route. The mean of maximum communication distance was 1444.34m, and surpassed the 1000m requirement.

Figure 5.16 plots the latency per WSM received by the vehicle , and it was observed that the mean of the maximum latency per vehicle was 0.018sec, considerably lower than the 1sec requirement.

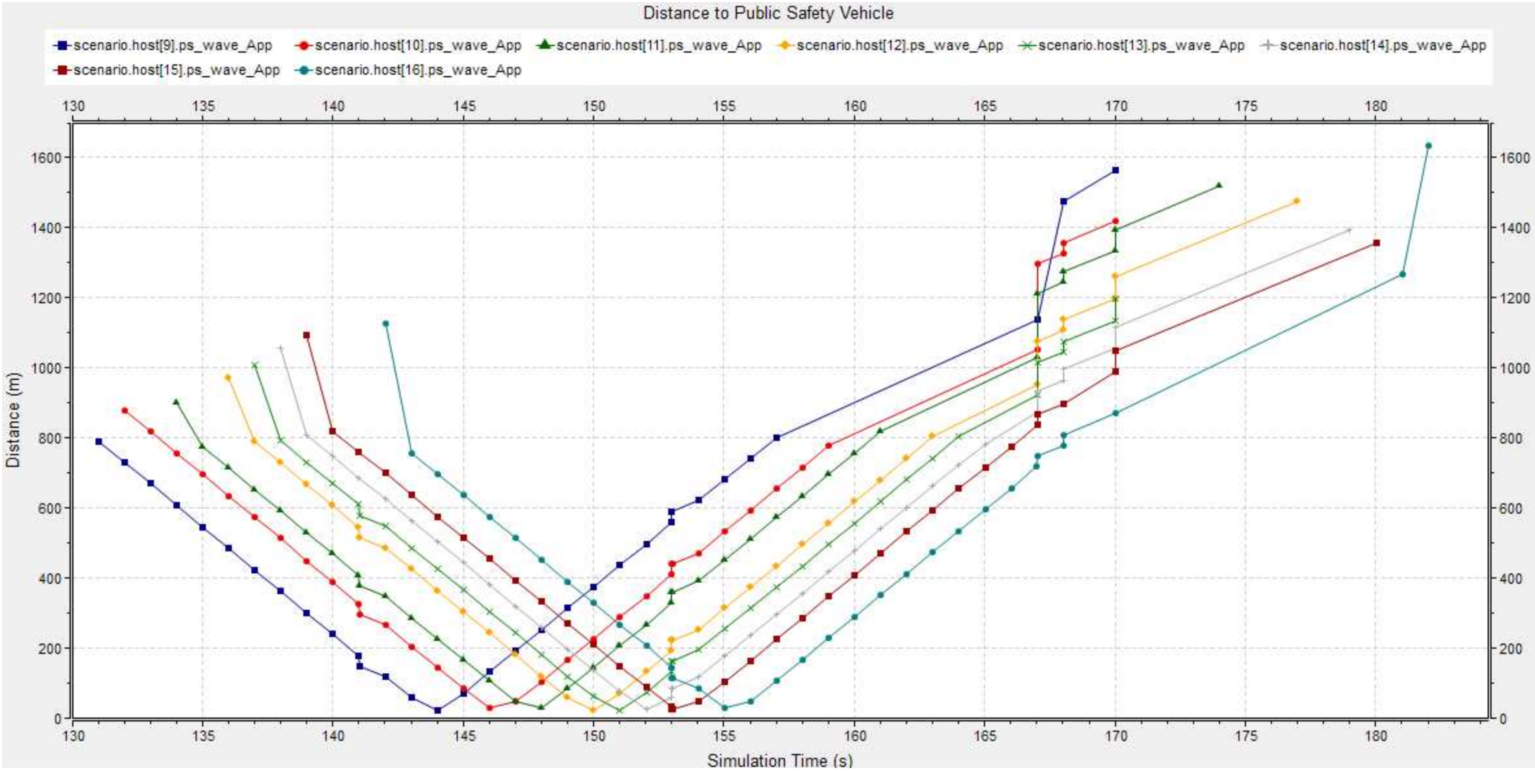

Figure 5.15: Highway: Distance to Public Safety Vehicle

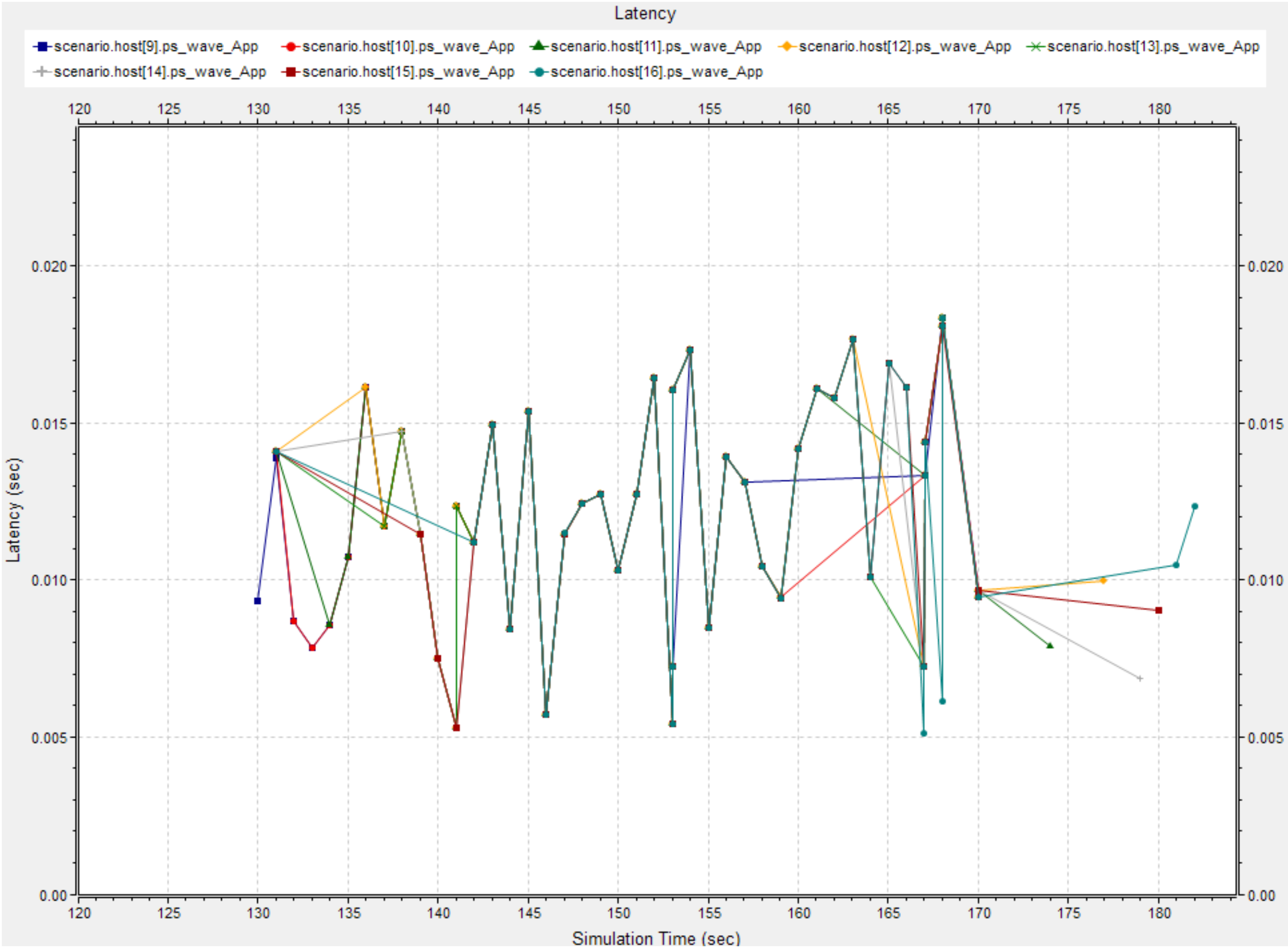

Figure 5.16: Highway: Latency - Approaching Emergency Vehicle Feature

Post Crash Warning

A total of eight vehicles were present in the scenario of which hosts labeled 10,11,12,13,14, and 15 approached the crashed vehicle in the opposite direction while hosts 5 and 6 were following the vehicle that crashed. . Figure 5.17 plots the distance to the crashed vehicle for every WSM received. The pattern observed here in this plot varied from the usual case as the vehicles in the opposite lane on a highway will not stop for an accident that has occurred in the opposite lane. But hosts 5 and 6 that were following the vehicle that crashed will still come to a halt and stay there. This can be spotted in the plot by looking at the two lines that run flat once they come closer to the crashed vehicle. The mean of maximum communication distance was 799.51m, and surpassed the 300m requirement.

Figure 5.18 plots the latency per WSM received by the vehicle , and it was observed that the mean of the maximum latency per vehicle was 0.019sec, considerably lower than the 0.5sec requirement.

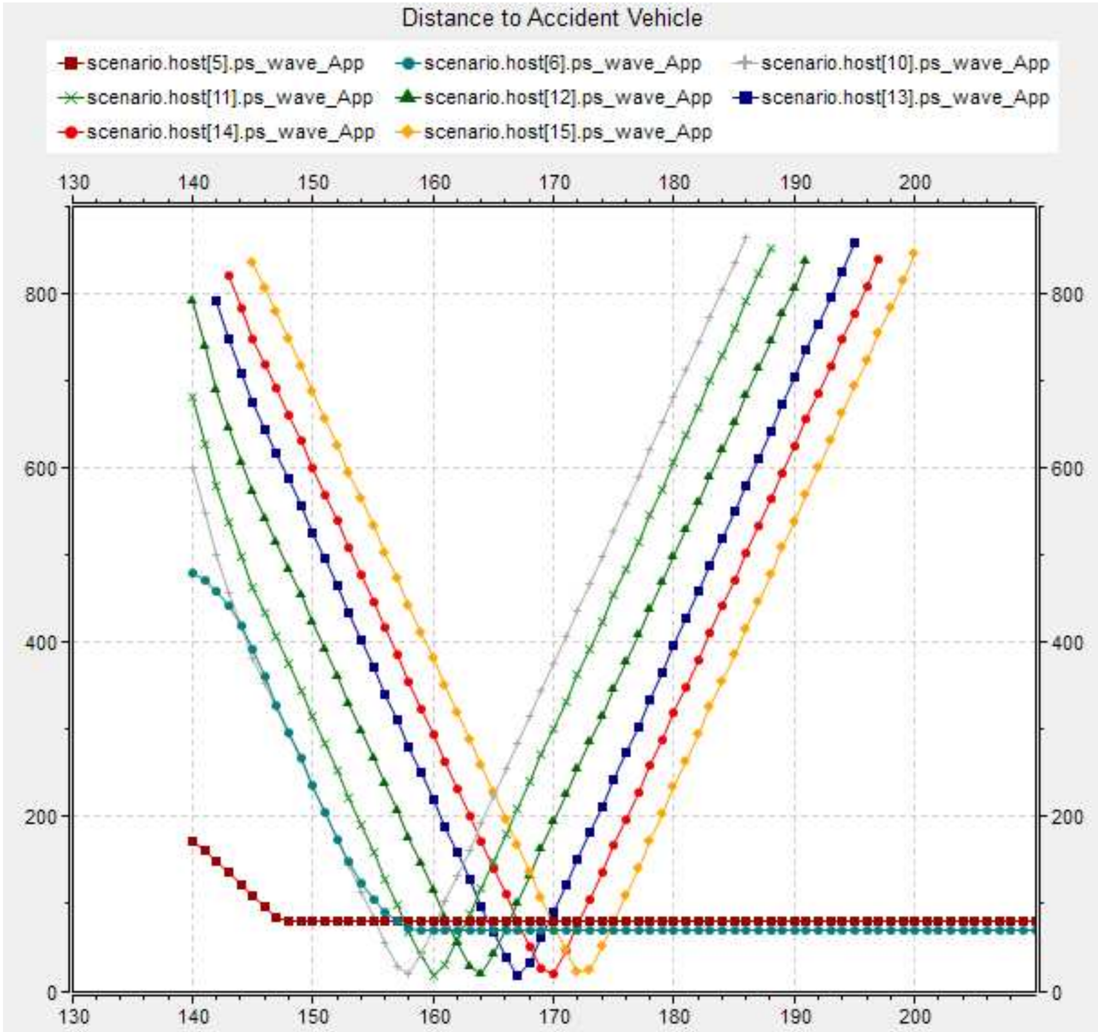

Figure 5.17: Highway: Distance to Accident Vehicle

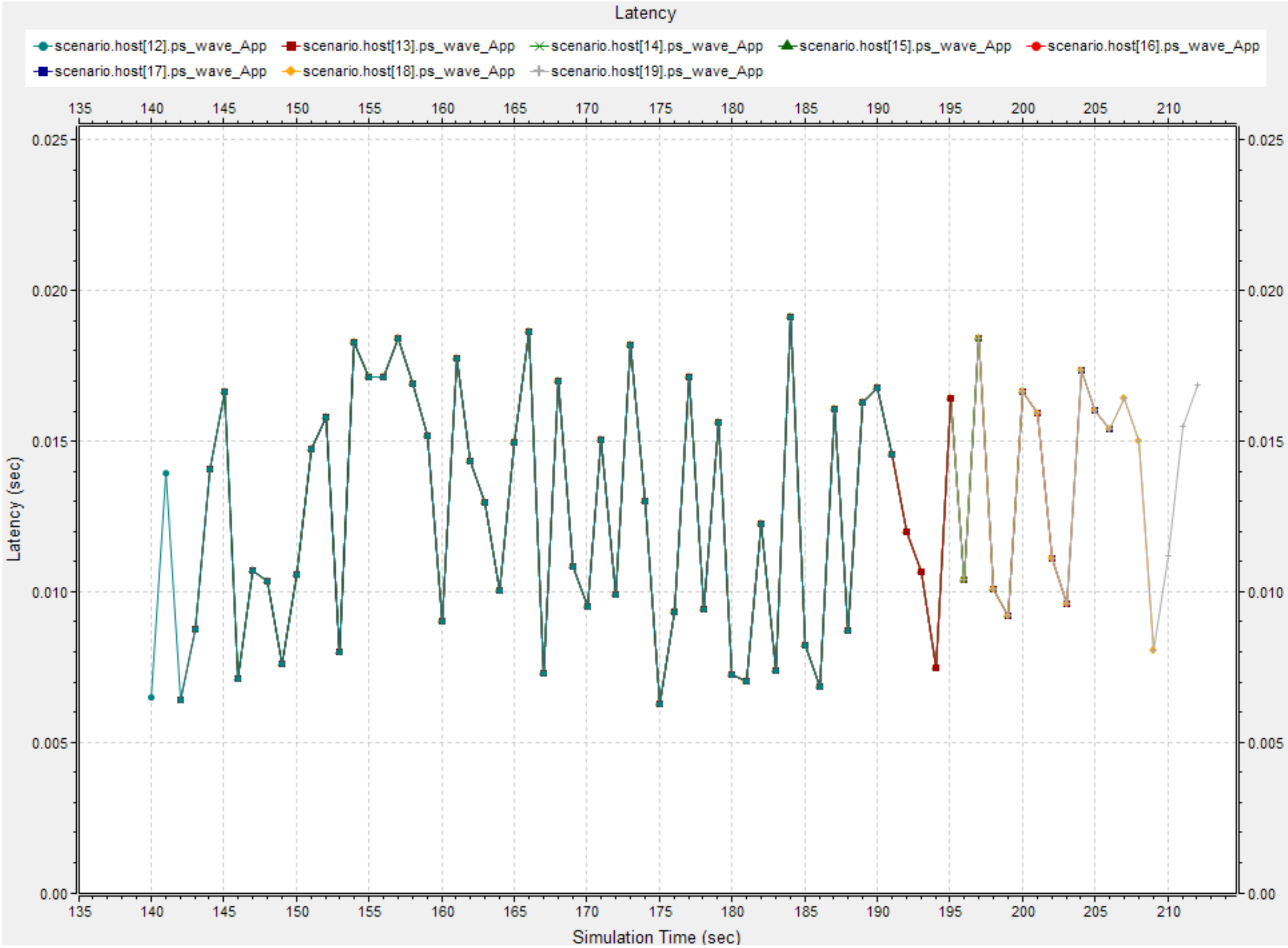

Figure 5.18: Highway: Latency - Post-Crash Warning Feature

Table 5.3 summarizes the results for the highway scenario.

Table 5.3: Highway: Summary of Results

Scenario	Mean of Max Communication Distance (m)	Mean of Maximum Latency (s)
Approaching Emergency Vehicle Alert	1444.34	0.018
Post-Crash Warning	799.51	0.019

Chapter 6

Future Work

The public safety application implemented in this thesis, is a good starting point for implementing a complete Intelligent Transport System consisting of many more applications with varied features all sharing the same network. Also, many other wired and wireless networks that support remote VANET applications can be added.

In direct relation to this thesis, in order to enhance the reach and functionality of the existing application features in certain cases, it will be beneficial to implement stationary road side unit which might run a different flavor of the application, helping retransmit the WSMs, hence increasing the range, and also if connected to a backbone network, can further enhance the reach of the information.

Traffic Lights could be implemented in the downtown scenario, and they could also be made intelligent in order to manipulate traffic flow in advance when they receive a WSM from an approaching emergency vehicle.

Intelligent dissemination of WSMs is another major research area concerning VANETs. The Application's range can be greatly increased, and also unwanted traffic on the network can be reduced by intelligently routing WSMs instead of using broadcast.

In the end, it would be of most importance to have a realistic VANET simulator which simulates traffic patterns and also all the applications and their effects on traffic flow. This will be needed once we have real world

VANETs in order to develop effective applications.

Bibliography

[1] Internet its consortium. http://www.internetits.org/en/top.html.

[2] The now: Network on wheels project. http://www.network-on-wheels.de.

[3] The prevent project. http://www.prevent-ip.org/.

[4] Vehicle safety communications project: Task 3 final report: Identify intelligent vehicle safety applications enabled by dsrc, 2005. This is an electronic document. Date of publication: [March, 2005]. Date retrieved: February 19, 2012.

[5] Ieee trial-use standard for wireless access in vehicular environments (wave) - multi-channel operation. *IEEE Std 1609.4-2006*, pages c1 –74, 2006.

[6] Ieee standard for wireless access in vehicular environments (wave) - networking services. *IEEE Std 1609.3-2010 (Revision of IEEE Std 1609.3-2007)*, pages 1 –144, 30 2010.

[7] W. Alasmary and Weihua Zhuang. The mobility impact in ieee 802.11p infrastructureless vehicular networks. In *Vehicular Technology Conference Fall (VTC 2010-Fall), 2010 IEEE 72nd*, pages 1 –5, sept. 2010.

[8] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. Sumo - simulation of urban mobility: An overview. In *SIMUL 2011, The Third*

International Conference on Advances in System Simulation, pages 63–68, Barcelona, Spain, October 2011.

[9] Pierpaolo Bergamo, Daniela Maniezzo, Kung Yao, Matteo Cesana, Giovanni Pau, Mario Gerla, and Don Whiteman. Ieee 802.11 wireless network under aggressive mobility scenario. In *in: Proc. International Telemetry Conference ITC/USA2003, Las Vegas*, 2003.

[10] D.N. Cottingham, I.J. Wassell, and R.K. Harle. Performance of ieee 802.11a in vehicular contexts. In *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, pages 854 –858, april 2007.

[11] Stefan Fischer, Jean-pierre Hubaux, Axel Wegener, Michal Pi, Maxim Raya, and Horst Hellbr. Traci: an interface for coupling road traffic and network simulators. *Proceedings of the 11th communications and networking simulation symposium*, pages 155–163, 2008.

[12] Sebastian Grafling, Petri Mahonen, and Janne Riihijarvi. Performance evaluation of ieee 1609 wave and ieee 802.11p for vehicular communications. In *Ubiquitous and Future Networks (ICUFN), 2010 Second International Conference on*, pages 344 –348, june 2010.

[13] M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE*, 7(4):12 –18, oct.-dec. 2008.

[14] J. Harri, F. Filali, and C. Bonnet. Mobility models for vehicular ad hoc networks: a survey and taxonomy. *Communications Surveys Tutorials, IEEE*, 11(4):19 –41, quarter 2009.

[15] Yong Shen, Qingquan Zou, and Zhimeng Liu. Performance evaluation

of ieee802.11-based ad-hoc wireless networks in vehicular environments. In *Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on*, pages 1 –4, sept. 2009.

[16] Jatinder Pal Singh and Nicholas Bambos. Wireless lan performance under varied stress conditions in vehicular traffic scenarios. In *Vehicular Technology Conference, 2002. Proceedings. VTC 2002-Fall. 2002 IEEE 56th , Volume: 2 , 24-28 Sept. 2002 Page(s): 743 -747 vol.2*, pages 743–747, 2002.

[17] C. Sommer, R. German, and F. Dressler. Bidirectionally coupled network and road traffic simulation for improved ivc analysis. *Mobile Computing, IEEE Transactions on*, 10(1):3 –15, jan. 2011.

[18] Christoph Sommer. Ieee 802.11p and ieee 1609.4 dsrc/wave models for the omnet++ mixim framework, January 2012.

[19] Christoph Sommer, David Eckhoff, Reinhard German, and Falko Dressler. A Computationally Inexpensive Empirical Model of IEEE 802.11p Radio Shadowing in Urban Environments. In *8th IEEE/IFIP Conference on Wireless On demand Network Systems and Services (WONS 2011), Poster Session*, pages 84–90, Bardonecchia, Italy, January 2011. IEEE.

[20] Christoph Sommer, Zheng Yao, Reinhard German, and Falko Dressler. Simulating the Influence of IVC on Road Traffic using Bidirectionally Coupled Simulators. In *27th IEEE Conference on Computer Communications (INFOCOM 2008): IEEE Workshop on Mobile Networking for Vehicular Environments (MOVE 2008)*, pages 1–6, Phoenix, AZ, April 2008. IEEE.

[21] Marc Torrent Moreno. Inter-vehicle communications: achieving safety in a distributed wireless environment. *thesis and dissertation*, 2007.

[22] R. Uzcategui and G. Acosta-Marum. Wave: A tutorial. *Communications Magazine, IEEE*, 47(5):126 –133, may 2009.

[23] Gongjun Yan. Bibliography on qos of ad hoc communications.

[24] Saleh Yousefi, Mahmoud Siadat Mousavi, and Mahmood Fathy. Vehicular ad hoc networks (vanets): Challenges and perspectives. In *ITS Telecommunications Proceedings, 2006 6th International Conference on*, pages 761 –766, june 2006.