“POLITEHNICA” UNIVERSITY OF BUCHAREST

COMPUTER SCIENCE DEPARTMENT

GRADUATION PROJECT

Information Dissemination in Vehicular Ad-hoc Networks

Scientific Coordinators:

Prof. Valentin Cristea, Ph.D, “Politehnica” University Bucharest
Prof. Liviu Iftode, Ph.D, Rutgers University, New Jersey, USA

Author:
Cristian Gorgorin

June 2006

TABLE OF CONTENTS

ABSTRACT

Vehicular Ad-Hoc Networks (VANET) are a particular type of wireless ad-hoc networks. They are formed when equipping vehicles on the roads with short-range wireless communication devices. Such an infrastructure could enable a wide range of applications, like dynamic route planning, or safety related applications. Most of them require a data dissemination model, which is a very challenging task in such a dynamic, large-scale environment. We have tried to address some of the difficulties of this problem. We have studied the performance of data forwarding algorithms, based on geographical routing. Testing is a difficult task with a vehicular network, because a large number of nodes are required in order to obtain relevant results. Simulation is therefore the only way to validate protocols in a VANET. We have developed an integrated simulator. It includes a complex mobility model for vehicular traffic and a wireless network simulation module. We have used our simulator in order to study the data forwarding protocols and we have come to the conclusion that, in sufficiently dense traffic scenarios, they can work. We have also estimated bandwidth requirements and came to the conclusion that advances in lower layers technology are still required, for a good performance of VANET protocols.

1. INTRODUCTION

Vehicle-to-vehicle communication is a concept greatly studied during the past years. Vehicles equipped with devices capable of short-range wireless connectivity can form a particular mobile ad-hoc network, called a “Vehicular Ad-hoc NETwork” (VANET). The existence of such networks opens the way for a large range of applications. We consider that 2 of the most important classes of such applications are route planning applications, and safety-related applications.

Route planning applications aim to provide drivers with real-time traffic information, which would, in the absence of a VANET, require expensive infrastructure. As opposed, the VANET approach is highly scalable and the maintenance costs are very low. Moreover, short-range wireless communication technologies (802.11) have no associated cost, other than the communication device.

Safety applications involve disseminating urgent information, which is unavailable in the driver’s field of view, or difficult to notice for reasons such as fog, other vehicles obstructing the line of sight and so on. For instance, a lot of accidents occur in foggy conditions, because drivers notice too late that some kind of incident has occurred in front of them. Safety at intersections could also be enhanced, because possible collisions could be detected in advance and the driver could be warned seconds before what would otherwise be an imminent accident.

Most applications to be deployed on top of a VANET require some sort of data-dissemination model. This is a challenging problem, due to the unique characteristics of a VANET. Such a network has a very high degree of mobility and a very large scale. Partitioning frequently occurs, making end-to-end communication impossible at times. Several studies [10] show that the performance of classical, topology-based routing protocols in vehicular networks is poor, due to the extremely high mobility of the nodes.

The focus of this document is on data-dissemination protocols, suitable for sending small amounts of information over the network. The main difference from common routing protocols is that forwarding decisions are made on a per-packet basis. Each node in the VANET must act as a router, but no hop-by-hop routes must be created; forwarding decisions about the next hop are made only when needed. Therefore, we adopt the terminology proposed in [1], which suggests using the term “data forwarding protocols” instead of “routing protocols”.

We do not address the problem of sending large amounts of information (for instance file transfer) to far-away nodes, as this seems impossible to achieve using only the ad-hoc network. Furthermore, we study applications which only require the identification of nodes by means of their location. Communicating with a specific node in the network, regardless of its geographical location, is also impossible to achieve using only the ad-hoc network.

Specifically, we are concerned with 2 different routing challenges: the query-reply problem, and posting a message. The query-reply problem is about a source node which issues a query message. The message must travel to a specific remote destination region, using only the ad-hoc network, collect some information from the nodes within that region, regardless of their identity, and return to the source. The posting of a message is done by a source, which intends to send a message to all or several of the nodes in a specific remote destination region, again regardless of their identity, and with no reply expected. We intend to prove that both these problems are feasible, in certain conditions, by using an inter-vehicular ad-hoc network.

The remainder of this document is organized as follows. Chapter 2 describes the simulation environment we have implemented and used in order to validate the studied protocols. Chapter 3 briefly describes the VITP protocol [2], an application-layer protocol with semantics similar to the ones required by the situations we study (query-reply and posting). In chapter 4 we describe in detail the query-reply problem and the routing solutions we suggest. The posting problem and the routing solutions we suggest are described in chapter 5. Chapter 6 presents some implementation details of the system we have developed. In chapter 7, we present all of the simulation results we have obtained, which lead to the conclusion that our solutions are feasible, and we also discuss some issues regarding bandwidth requirements. Chapter 8 describes some real-world applications which could be implemented by using our protocols and we also comment on security issues in VANETs. We conclude in Chapter 9.

2. SIMULATION ENVIRONMENT

The simulation of a VANET requires 2 different components: a network simulator, capable of simulating the behavior of a wireless network, and a vehicular traffic simulator, able to provide an accurate mobility model for the nodes of a VANET. Recent studies [3] have proven that the vehicular mobility model is very important, and in order to obtain relevant results, the two components should be integrated. If an inaccurate mobility model is used, like the popular random waypoint model (which may work for some mobile ad-hoc networks, but is definitely not an accurate representation of mobility in a VANET), false results can be obtained [3][4]. However, we believe the mobility model implemented in [3] or [4] is still not an accurate representation of real vehicle mobility. Thus, the authors of [4] use real maps, in the TIGER format [5] and their vehicles move along the streets. But each vehicle moves completely independent of other vehicles, with a constant speed randomly chosen. Multi-lane roads or traffic control systems are not taken into consideration. The mobility model in [3] is more complex. It also uses TIGER files, and considers car-following models. Thus, the motion of a vehicle is influenced by the preceding vehicle. They also implement traffic control systems: timed traffic lights and stop signs. However, multi-lane roads are not taken into consideration.

We have chosen to develop our own simulation tool, comprising the 2 previously mentioned components: a microscopic traffic simulator, and a wireless communication model. The entire project was implemented in Java. We have also implemented a graphical user interface, using OpenGL for Java (JOGL). However, during most of the simulations we have performed, we disabled it, in order to improve the simulator’s performance.

2.1 SYSTEM ARCHITECTURE

The simulator we have developed is a discrete event-based simulator. The simulation time advances with a fixed time resolution after executing the application code for the current moment of the simulation time. More specifically, at every moment of the simulation time, all the current events are pulled from a queue of events, and handled in a random order.

The events queue can hold three types of events: send, receive or GPS. A send event for a specified node triggers the calling of the node's handler procedure responsible for preparing a message. It also schedules the corresponding receive event(s) for the desired receiver(s), according to the network module of the simulator that decides who should receive the message. The receive event is associated either with a node, or with a group of nodes (broadcast) and it calls the appropriate handler in each of the receiving nodes. The GPS event is scheduled at a regular time interval for each node, in order to simulate the way a real VANET application collects GPS data periodically.

Besides these three types of events, the mobility module updates periodically, with a specific time resolution, the position of each node, which is a vehicle, according to the vehicular mobility model.

2.2 MOBILITY MODEL

2.2.1 MAPS

It is clear that a digital map is a basic requirement in order to consider any kind of VANET application. Each vehicle which is part of the system should have such a digital map. For our simulator and application, we have chosen to use TIGER files [5], which are freely-available, real digital maps of the USA. The TIGER files contain detailed geographical information about all the roads in a region, from large highways to small streets. The data they contain comes in the form of geographical coordinates (latitude, longitude) for the roads. Thus, for every road, the TIGER files specify its end points, along with as many intermediary points as needed. If the road is straight, no intermediary points are contained, because they can easily be computed through interpolation. If the road has curves, a large number of intermediary points are represented, thus providing an accurate map. Furthermore, for each road a simple information is given, indicating its "class" (whether it is a small street, a local road, a State Route, an Interstate Highway and so on).

However, the TIGER database unfortunately lacks other traffic-specific information, like the number of lanes, or traffic control systems (traffic lights, yield or stop signs). We believe that a mobility model which does not take multiple lanes or traffic control systems into consideration is not realistic enough; therefore we have added some extra information, based on simple heuristics and based on the road class information included in the TIGER files. Some of the rules we have used include more lanes for higher class roads, yield or stop signs for lower class roads, traffic lights between equally important roads, longer green period for the road with the higher number of lanes and so on. The heuristic algorithm has been combined with a random factor, in order to obtain various scenarios.

2.2.2. MICROSCOPIC TRAFFIC SIMULATOR

A traffic simulator which takes into account the actions of each individual vehicle is a microscopic simulator, as opposed to macroscopic simulators, which describe the evolution of traffic using global measures, like flow or traffic density. Macroscopic simulators can be used to better understand the traffic dynamics and to better design traffic-related facilities (traffic lights, number of lanes, lane closures and so on). However, a much higher level of detail is necessary for the study of a vehicular network; therefore we have developed a microscopic traffic simulator. It is based on the driver behavior model developed by Wiedemann [6][7]. The same model is used in the widelyused commercial traffic simulator “VISSIM” [8]. Like many other vehicular traffic simulators, VISSIM’s purpose is modeling and forecasting vehicle traffic flow, for decisions like adding a new lane, studying the impact of lane closures on traffic, building an overpass and so on. Such simulators are difficult to integrate with network simulators, especially since most of them are commercial products.

Next, we briefly describe the driver behavior model we have implemented, which is based on [9]. The basic idea of Wiedemann is the assumption that a driver can be in one of four modes: free driving, approaching, following or braking.

Free driving means there is no influence from preceding vehicles on the same lane. In this situation, the driver will seek to obtain and maintain a desired speed. The desired speed and the acceleration depend on the driver personality, and on the road characteristics.

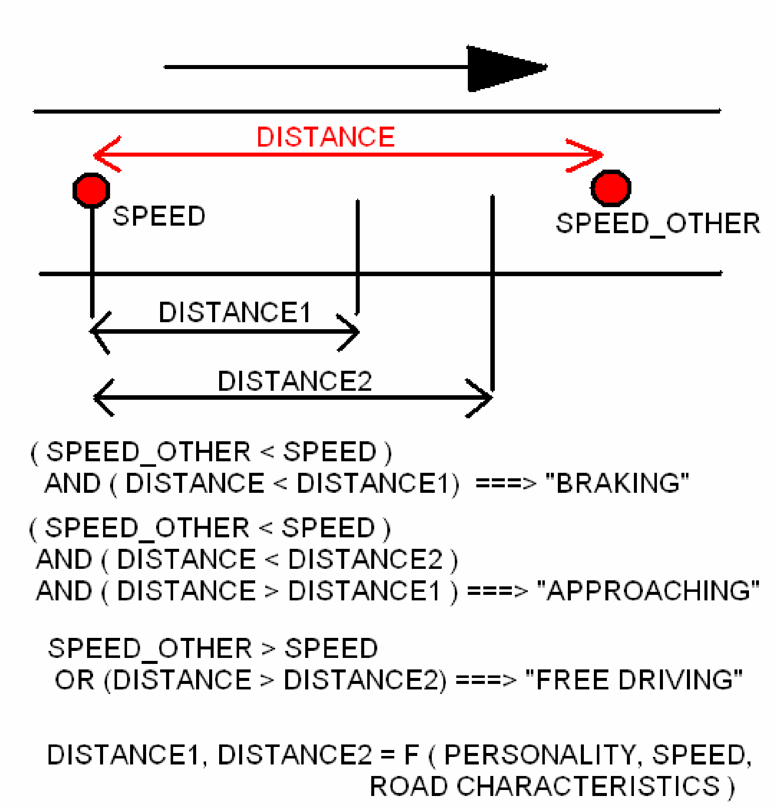
The “approaching” mode means that there is a slower, preceding vehicle which influences the driver. In this situation, she/he will apply a deceleration in order to obtain the same speed as the preceding vehicle. The deceleration is a function of the distance between the two vehicles, their speeds, as well as other parameters.

The “following” mode means there is a preceding vehicle, but the speeds of the two vehicles are practically equal. In this situation, the driver will seek to keep the speed constant.

The “braking” mode means there is a slower preceding vehicle, very close in front. In this mode, due to the immediate danger, the driver will apply high deceleration rates.

Figure 1 presents some basic rules to determine the mode a driver is in. Thus, there are 2 thresholds, “distance1” and “distance2” according to the notation in the figure. If the preceding vehicle is closer than “distance1” and slower than the current vehicle, then the latter will be in “braking mode”. If the slower, preceding vehicle is between “distance1” and “distance2” in front, then the mode will be “approaching”, and the current vehicle will gradually decelerate. If the preceding vehicle is further away than “distance2”, then it does not influence the current vehicle in any way, and it will be in the “free driving” mode. These thresholds (“distance1” and “distance2”) are not constant, but they depend on the driver’s personality and on the vehicle’s speed.

We have also implemented a lane-changing model, for multi-lane roads. The model we have implemented is based on the lane-usage rules valid throughout most part of Europe. Thus, the usage of the first lane is required, unless it is occupied. That means that a driver will always try to stay on the lower lanes, except when passing another slower vehicle. Passing on the right is not allowed. These rules are not valid in city environments, near intersections, where lanes are selected based on the direction the driver intends to follow.



```
( SPEED_OTHER < SPEED )
 AND ( DISTANCE < DISTANCE1)  ===> "BRAKING"
( SPEED_OTHER < SPEED )
 AND ( DISTANCE < DISTANCE2 )
 AND ( DISTANCE > DISTANCE1 ) ===> "APPROACHING"
  SPEED_OTHER > SPEED
   OR (DISTANCE > DISTANCE2) ===> "FREE DRIVING"

  DISTANCE1, DISTANCE2 = F ( PERSONALITY, SPEED,
                            ROAD CHARACTERISTICS )
```

Figure 1 - Driver modes

The lane-changing model we have designed and implemented is based on the hierarchy between the 4 driving modes. Whenever a driver is in a different mode than "free driving", she/he will always check if the higher lane can provide a superior mode. If that is the case, the driver will switch to a higher lane. Similarly, whenever a driver is in a different mode than "braking", she/he will always check if the lower lane provides at least similar conditions. If that is the case, the driver will switch to a lower lane. The order of these checks is important. The higher lane is checked first. Thus, if a driver uses lane 2 and approaches another slower vehicle, it will first check if lane 3 is empty, and if that is the case it will switch to lane 3 (only if it can safely complete the switch, without interfering with any vehicles approaching from behind). If it had first checked the lower lane, it could have discovered that it is empty and it would have decided to use lane 1 for passing the vehicle on lane 2, which is forbidden in most European countries. However, it is not forbidden in the United States, where any lane can be used for passing. US traffic could easily be simulated, by making a random decision whether to first check the higher lane or the upper lane when looking for superior driving conditions.

We have also incorporated traffic control systems into our driver behavior model. Thus, the vehicles we simulate are aware of traffic lights, priority roads and "yield" or "stop" signs, and their motion is simulated according to these traffic control systems.

Different classes of drivers (aggressive, regular, calm) can easily be modeled by using the numerous model parameters. Each driver class is represented by a certain set of values for the parameters. In order to further differentiate the drivers, there is also a small deviation from the specified values, deviation computed randomly for each driver.

The authors of VISSIM prove that the model is accurate, by comparing simulated traces with real measurement data taken from a German freeway and from a US freeway [9]. Still, the model is supposed to be accurate not only for freeway conditions, but also for city-like scenarios. To further calibrate and validate our model, we have focused on a simple, yet very frequent, city-like scenario. We consider a typical intersection where vehicles are queued, waiting for a red traffic light to become green (see Figure 2).

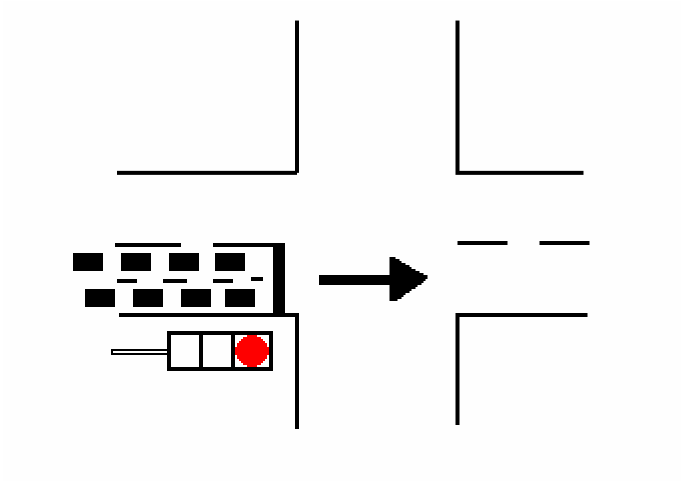
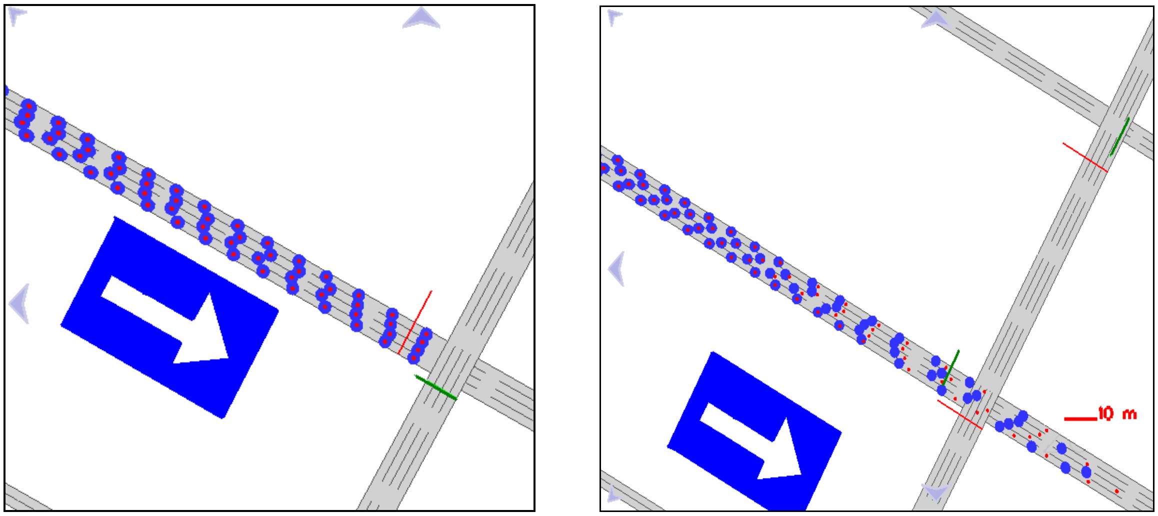


Figure 2 - Typical city scenario

We assume that all vehicles intend to drive forward. Let “FlowPerLane” be the number of vehicles that pass the intersection per second, per lane, during a time period beginning immediately after the light has become green. We consider “FlowPerLane” to be a very important parameter characterizing the motion of vehicles through the intersection, because it is influenced by several parameters of our driver behavior model, like vehicle acceleration, desired distance from the preceding vehicle, or reaction time.

We have chosen the intersections “Piata Victoriei” and “Arcul de Triumf” in downtown Bucharest for measurements. Both intersections meet the above mentioned assumptions. The number of lanes is large (4, respectively 3), and all vehicles are required to drive forward. We measured FlowPerLane by counting passing vehicles during the green phase of the traffic light. We repeated the experiment several times, varying the time frame during which we counted the vehicles, because we suspected there might be a difference between the flow values at the beginning and towards the end of the green phase. The results, however, did not indicate such a difference.



Figures 3 a,b – Simulation screenshots

We simulated a similar intersection using our driver behavior model. Figure 3 shows screenshots of our simulator's JOGL GUI, taken during the simulation. Figure 3a shows the vehicles still waiting for the red light to become green, while Figure 3b shows the vehicles as they have started passing, as the light has turned green. We have calibrated some of the numerous driver behavior model parameters, based on the real results obtained. Finally, with the calibrated parameters, we have performed several measurements. The measured data and the simulated data are presented in the table in Figure 4.

It is easy to notice the similarity between the simulated data and the real situation. The simulated data values have an average of **0.46** and a standard deviation of *0.03*. The measured data values from "Piata Victoriei" have an average of **0.45** and a standard deviation of *0.04*. Finally, the measured data values from "Arcul de Triumf" have an average of **0.47** and a standard deviation of *0.02*.

SIMULATED DATA				REAL MEASURED DATA			
NUMBER OF VEHICLES	NUMBER OF LANES	TIME (SECONDS)	FLOW PER LANE (VEHICLES/SEC)	NUMBER OF VEHICLES	NUMBER OF LANES	TIME (SECONDS)	FLOW PER LANE (VEHICLES/SEC)
47	4	26	**0.45**	PIATA VICTORIEI			
46	4	26	**0.44**	12	1	26	**0.46**
49	4	26	**0.47**	12	1	26	**0.46**
49	4	26	**0.47**	10	1	26	**0.38**
48	4	26	**0.46**	49	4	26	**0.47**
51	4	26	**0.49**	50	4	26	**0.48**
47	4	26	**0.45**	6	1	12	**0.50**
48	4	26	**0.46**	18	4	12	**0.38**
25	4	12	**0.52**	23	4	12	**0.48**
24	4	12	**0.50**	22	4	12	**0.46**
22	4	12	**0.46**	ARCUL DE TRIUMF			
24	4	12	**0.50**	58	3	40	**0.48**
24	4	12	**0.50**	55	3	40	**0.46**
65	4	40	**0.41**	53	3	40	**0.44**
67	4	40	**0.42**	57	3	40	**0.48**
67	4	40	**0.42**	37	3	26	**0.47**
68	4	40	**0.43**	39	3	26	**0.50**
66	4	40	**0.41**	35	3	26	**0.45**
69	4	40	**0.43**	38	3	26	**0.49**

Figure 4 - Comparison of measured and simulated data

Based on the clear similarities between the real and the simulated data, we conclude that the model is an accurate approximation of vehicular mobility, in the above-mentioned, frequently-met, city scenario.

2.2.3 IMPLEMENTATION DETAILS

The mobility module is concerned with moving all vehicles on realistic trajectories. We have implemented a complex mobility model, including lane changing, car-following, and traffic control systems.

The module uses a representation of a map, which contains information about all the roads within a specific region. The module must also keep track of all the vehicles within the region and simulate their motion, based on a complex model. Thus, a vehicle will move depending on the traffic conditions around it, specifically on the position of adjacent vehicles. Given one chosen vehicle, a fast, efficient way of obtaining the characteristics of the adjacent vehicles was necessary. An exhaustive search, through all vehicles, in order to see which ones are near-by, was definitely not a scalable approach. Thus, we chose to keep the vehicles on each road in a sorted array. The array is sorted by position (offset from the road's starting point), thus allowing fast retrieval of adjacent vehicles, for any given vehicle. A simplified UML class diagram is shown in Figure 5.

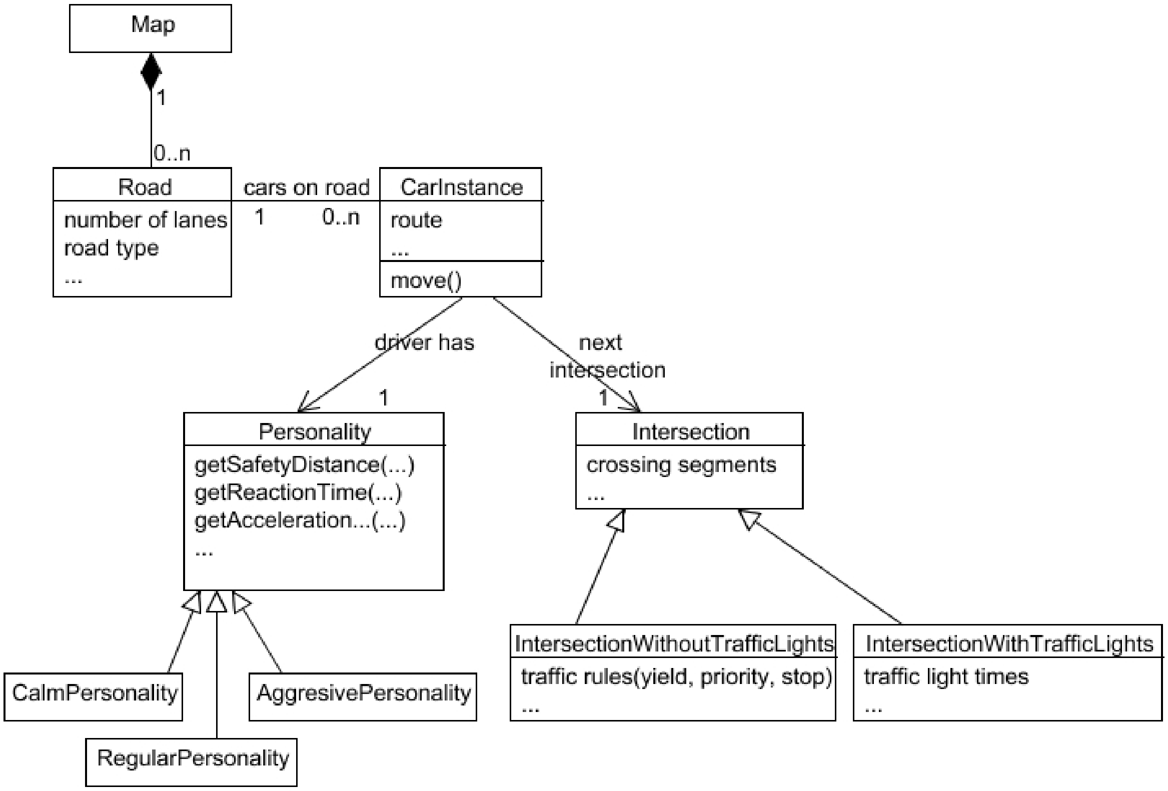


Figure 5 – UML class diagram for the mobility module

Thus, the map contains an array of roads. Each road contains an array of vehicles. A vehicle is modeled through the "CarInstance" class, which has a method called "move". That method takes care of the entire motion-related logic. It basically computes a new position for the vehicle by simulating its motion during a small time frame (we have used 10 milliseconds). This method is called by the simulator engine every time frame, for each vehicle. The motion is basically dependent on 3 elements: traffic control systems, the position of adjacent vehicles, and the driver's personality.

A vehicle will first check traffic control systems present at the intersection ahead. Thus, whenever a vehicle enters a new road segment, a reference to an Intersection object is computed. That object represents the next intersection through which the vehicle will pass. When the vehicle will be within a specific range from the intersection, it will start making decisions based on specific traffic regulations for that intersection (regulations which can be retrieved from the Intersection object). For instance, if it is an intersection with traffic lights and the light is red, the vehicle will start decelerating. It will also decelerate if the intersection has no traffic lights, but the vehicle is approaching it on a road with a "yield" or "stop" sign. The vehicle also determines the lanes it should drive on, if a turn is going to be made at the intersection.

After decisions based on traffic control systems are computed, the position of adjacent vehicles is considered. That can be quickly retrieved from the sorted array held in the "Road" class, without having to search through the entire array. Depending on that, a vehicle can determine the driving mode it is in (free driving, approaching, following or braking – see Section 2.2.2). It can also make lane-changing decisions, if other lanes offer better traffic conditions, but decisions taken in the first stage (related to traffic control systems) will not be overridden. For instance, if the decision to decelerate was made when considering traffic control systems, then the vehicle will definitely not accelerate, even if adjacent traffic permits such a decision. Another example is a vehicle which intends to turn right at the intersection ahead and has determined that the usage of the first lane is required. Such a vehicle will not choose to switch to higher lanes when approaching the intersection, even if those higher lanes have more fluent traffic.

Every step of the motion-related logic (including analysis of traffic control systems and analysis of adjacent vehicles) is influenced by the driver's personality, modeled by an instance of the Personality class, to which each vehicle holds a reference. The Personality class encapsulates a large number of parameters. Different personalities can be easily modeled, by setting the parameter values accordingly.

Each time frame, after the simulator engine calls the “move()” method of all the CarInstance objects, it takes care of re-sorting the array of vehicles kept in each “Road” object.

2.3 NETWORK SIMULATOR

The network simulator module simulates delivery of messages from one node to another. At this level, messages are seen as sequences of bytes. Each node in the network has a unique identifier, its “address”. Basically, this simulates MAC-layer transmission, and there is no support for higher layer protocols. There are 2 types of messages, unicast, when the identifier of the destination node is specified, and broadcast, when all nodes in the wireless range will receive the message. Basic 802.11 rules are implemented by our network simulator module.

In fact, addressing in a real vehicular network is a difficult challenge. In a classical, fixed network, the IP layer was introduced because the flat addressing scheme provided by the MAC layer was not scalable. IP addressing is hierarchical, and that offers benefits like route summarization, with the purpose of reducing routing tables and therefore making routing in very large scale networks (like the Internet) possible. In a vehicular ad-hoc network, there is no hierarchy between the nodes, because the topology changes extremely fast; therefore a supplementary routing layer on top of the flat addressing scheme offered by the MAC layer is not necessary. Of course, this flat addressing scheme means that classical routing algorithms, based on routing tables, will not work, because of the immense size of such routing tables.

Flat addressing means that each node has a unique address. There are 2 possibilities: the address could be fixed (once established for a vehicle, it would never change again) or variable. If a variable address solution is chosen, a way of assuring the uniqueness of addresses has to be developed. Moreover, difficult security issues appear if there is no way of identifying nodes in the network (see Section 7). If a fixed address solution is chosen, privacy becomes an issue. At the present moment, each vehicle has a license plate, which basically reveals the identity of a vehicle to every person/camera which actually sees the plate. With the deployment of a vehicular network which uses fixed addresses, the identity of a vehicle would clearly be revealed to every vehicle in its wireless range. That may not be considered a severe intrusion, but what would happen if the identity of a vehicle (together with other information, like location or speed) would be revealed to every node in the entire vehicular network? That would definitely not be acceptable. Therefore, a method to ensure that vehicle identity is protected is required. Such issues are outside the scope or our document. Therefore, without loss of generality, we have considered fixed addresses for the vehicles in our simulator.

3. VITP – APPLICATION LAYER PROTOCOL

The Vehicular Information Transfer Protocol (VITP) [3] is an application-level protocol which relies on the ability of the underlying VANET to transport messages. VITP messages can be one of 2 types: GET or POST. GET messages can be further classified as requests or replies. Every VITP message has an origin and a destination, specified from a geographical point of view. The VITP specification describes both the source and the destination of a message as a pair of identifiers: (road_id ; segment_id). Although this may seem like a limiting approach, we consider it does not affect the fundamental ideas behind the protocols we study and therefore we adopt the same convention. However, alternate representations of a region (like, for instance, a point and a circular radius, or even the modeling of more complicated perimeters) could be required for the implementation of some applications.

A GET request is a message originating from a (possibly human) user, trying to find out certain information. The request must be routed by the VANET towards the destination region. There, it must visit a number of nodes (regardless of their identity) and collect information from them. A return condition can be specified. For instance, the request could be considered satisfied after it has visited N nodes within the destination region, or after a timeout has expired. When a node within the destination region receives the message, it rewrites it, typically appending its own useful information, depending on the specific query the message raises. When the return condition is met, the message is transformed into a GET reply, which must be routed by the VANET back towards the source. This time, identity is important, because the same node which has originated the query is the only one interested in the reply.

A POST message also has a destination region, towards which it must be routed. There, it must get to as many nodes as possible within the destination region (regardless of their identity). Unlike the GET messages, no reply is expected by the source. The role of POST messages is to deliver information to nodes in a destination region, instead of query for information.

We consider VITP to be a protocol suitable to model the interaction required by the applications this document is focused on. Therefore, we adopt the terminology used by the authors – GET, POST, request, reply.

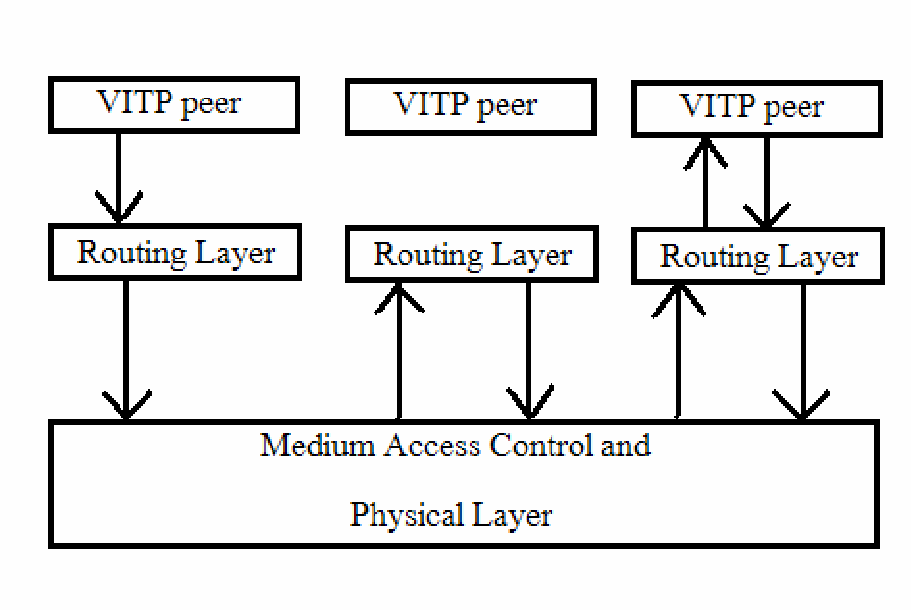


Figure 6 - Protocol layers

Still, while VITP is just an application-layer protocol, our focus is on the routing details which VITP requires in order to function properly over a VANET.

4. ROUTING LAYER CHALLENGES AND SOLUTIONS

4.1 QUERY-REPLY

It has been proven [10] that the very high mobility of a VANET makes the use of classical MANET protocols impossible. Thus, routing protocols which attempt to maintain routing tables (pro-active protocols) do not work, because the neighbors change very rapidly, and the overhead required to maintain tables in such a large scale network is immense. Classical protocols which try to establish a route when it is required (re-active protocols), like AODV, also experience problems, because end-to-end routes (specified as a sequence of hops) only last for very short time periods, due again to the very high mobility.

The alternative is to use location-based routing, which has been suggested as a good approach for vehicular networks [11]. The applications we study require a geographical data dissemination mechanism, which is not concerned with the identity of the intermediate nodes, but only with the final geographical destination of a packet. Thus, forwarding decisions are made on a “per-packet” basis [1], always trying to move the packet towards its geographical destination.

The possible partitioning of vehicular networks gave birth to another concept – opportunistic forwarding. Under this approach, messages are stored at intermediate nodes, when no suitable next hop exists, with the hope that such a node will appear in the near future [12]. Simulation results show, as expected, that delays due to network partitioning when using opportunistic forwarding are huge (the order of 100 seconds) compared to delays due to the wireless propagation of a message [12] [13]. For the query-reply problem, such large delays raise a particular problem: the source of the query can move quite a lot in the meanwhile and the reply will be impossible to deliver, as the source’s location becomes unknown. We consider it more useful to send a reply immediately to the source, which announces the partitioning and its location. The source of the query can try re-issuing it, if the response is not satisfactory.

For the geographical forwarding to work, we make the assumption that vehicles broadcast periodic beacon messages every second, announcing their position. Using these beacons, each vehicle is always aware of the location of its neighbors, which it can use for the routing of VITP messages. To make the simulation of common GPS devices more realistic, in our simulator vehicles only get to know their real position every second, and do not use any interpolation methods to update their known position throughout the 1-second interval. Still, it is worth noting that more expensive GPS devices exist, which use complex methods in order to provide continuous location information. The use of such devices could enhance the performance of the studied protocols.

However, it must be noted that no guarantees about the delivery of messages can be made, if only using the ad-hoc network based on short-range wireless connectivity. Each message is sent on a "best-effort" basis, but a failure probability, due to network partitioning, will always exist. Guarantees could only be made if alternative communication solutions are used, like 3G wireless communication. In this document we only address routing in pure ad-hoc networks.

In the following sections, we study separately 2 very different scenarios: traffic on highways and traffic in city environments.

4.1.1 HIGHWAY SCENARIO

For a highway, examples of applications which could use such a protocol include: "what is the average speed of at least N vehicles on section X of the highway?" or "where is the next gas station? what are the prices?".

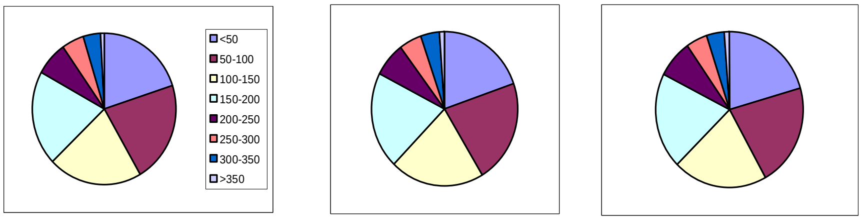
The natural routing solution to use for a highway scenario is pure geographical routing, because highways are pretty straight and the problem of dead-ends does not exist. The message is supposed to travel from node to node to a remote destination region. If every node along the path broadcasted the message, the result would be severe network flooding. A solution based on broadcast is clearly not scalable. However, using unicast raises another problem. Each node must be aware of its neighbors, along with their location. While this can be accomplished with regular beacon messages, the exact neighbor location can unfortunately not be known, due to 2 limitations: GPS error and very high mobility. The beacons period cannot be very small, because it wastes bandwidth. We must also consider an expiration period, a time interval after which, if no beacons are received from a neighbor, it is considered to be out of the wireless range and eliminated from the neighbors list.

We decided to perform simulations in order to verify if the approach is viable. We simulated a typical highway traffic scenario. We chose a 1-second time interval for the beacons and a wireless range of 200 meters. Vehicles were chosen randomly at random moments and the information they had about their neighbors was checked against the real situation. We varied the expiration period, the number of lanes and the penetration ratio (the percentage of vehicles equipped with the system). We performed simulations with 2-way traffic and 1-way traffic. The results are presented in the tables in Figure 7.

The conclusions which must be drawn refer to how many neighbors are actually outside of the wireless range (set at 200 meters). We will refer to such neighbors as “obsolete neighbors”. As expected, with a lower expiration period, the number of obsolete neighbors is lower, because they are eliminated sooner from the neighbors list. It can also be noticed that 1-way traffic generates a lower number of obsolete neighbors. This is also expected, because relative speeds are much higher with 2-way traffic, therefore neighbors are more likely to move out of the wireless range before they are eliminated. On the other hand, no significant influence of the number of lanes or the penetration ratio can be observed.

EXPIRATION PERIOD = 5 seconds

	60%		80%		80%	
DISTANCE FROM	3LANES		3LANES		5LANES	
NEIGHBOR	2WAY		2WAY		2WAY	
(meters)	(count)	%	(count)	%	(count)	%
<50	388912	**19.96**	579727	**19.81**	1586864	**20.54**
50-100	423533	**21.74**	630818	**21.56**	1651202	**21.37**
100-150	408705	**20.98**	607695	**20.77**	1596177	**20.66**
150-200	398091	**20.43**	595917	**20.37**	1550681	**20.07**
200-250	142040	**7.29**	210572	**7.20**	584362	**7.56**
250-300	95718	**4.91**	145349	**4.97**	366357	**4.74**
300-350	74539	**3.83**	123039	**4.21**	298249	**3.86**
>350	16582	**0.85**	32779	**1.12**	93056	**1.20**



EXPIRATION PERIOD = 3 seconds

	60%		80%		80%		60%	
DISTANCE FROM	3LANES		3LANES		5LANES		3 LANES	
NEIGHBOR	2WAY		2WAY		2WAY		1WAY	
(meters)	(count)	%	(count)	%	(count)	%	(count)	%
<50	306385	**21.95**	506136	**21.90**	1519171	**22.57**	98080	**22.18**
50-100	330357	**23.67**	548460	**23.73**	1590778	**23.64**	114453	**25.89**
100-150	321261	**23.02**	528069	**22.85**	1531603	**22.76**	109074	**24.67**
150-200	312777	**22.41**	514977	**22.28**	1489151	**22.13**	111034	**25.11**
200-250	96006	**6.88**	161866	**7.00**	466352	**6.93**	9498	**2.15**
250-300	28538	**2.04**	51017	**2.21**	132202	**1.96**	0	**0.00**
300-350	288	**0.02**	522	**0.02**	1292	**0.02**	0	**0.00**
>350	1	**0.00**	7	**0.00**	4	**0.00**	0	**0.00**

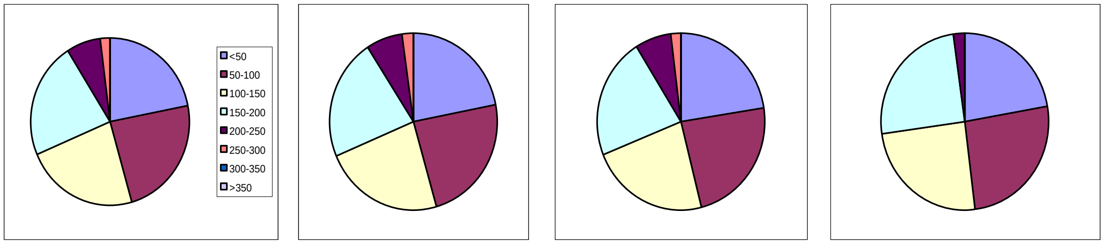
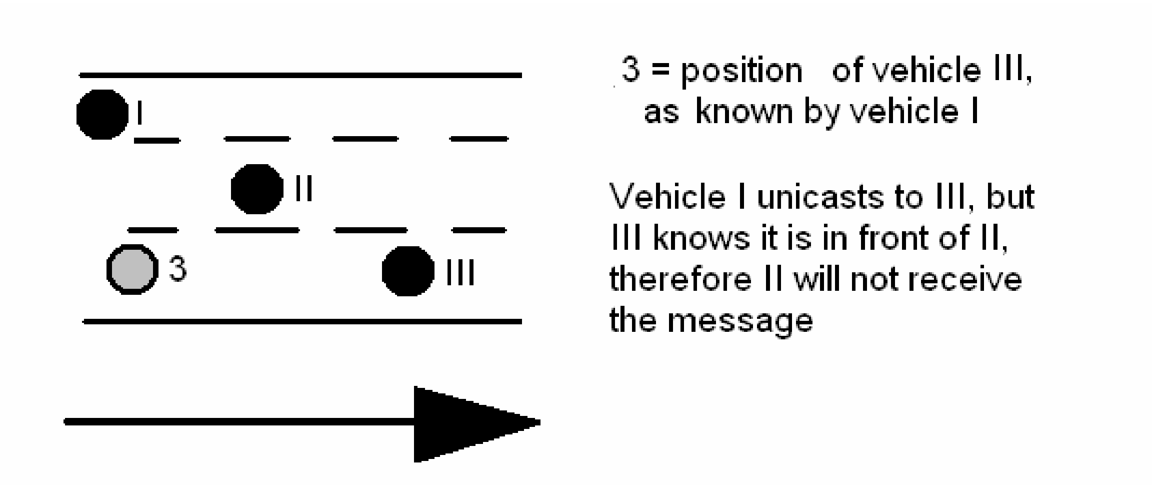


Figure 7 - Accuracy of neighbor information – highway scenario

Based on the results, we decided to consider a 3-seconds expiration period for any further simulations. For the 2-way traffic scenario, the results show a percentage of fewer than 10% obsolete neighbors. While this is a promising figure, a multi-hop routing protocol which does not take this source of failure into account cannot succeed. For instance, if a packet must go through 20 intermediate nodes until the destination and at every node, it has a 10% chance of failure (due to obsolete neighbors), then the delivery probability will be $(0.9)^{20}$, which is extremely low. Because obsolete neighbors cannot be totally eliminated, we conclude that a routing protocol based on unicast cannot work unless it uses some sort of acknowledgement for each sent message. If the acknowledgement does not arrive, a node will choose another neighbor to which it forwards the message.

Another difficult problem for the VITP query mechanism is how to get the message to pass through each node in the destination area. The naïve approach is to always unicast to the “next” node (the closest one). However, this information is also based on the neighbors which every node is aware of. As previously discussed, this information is not always perfectly in accordance with the reality. Thus, the possibility that a node is “skipped”, because its neighbors do not know its exact location, appears. For instance, in Figure 8, the message is supposed to pass from I to II, and then from II to III, but II will be skipped, because I believes that the order is I, III, II, and therefore will unicast the message to III. If the application does not require that the message visits every node in the destination region, but only some of them, the approach is viable. However, there may be applications for which skipping a node is not acceptable. For instance, for the “gas-station” query previously mentioned, if the skipped node is the gas-station itself, then the query will not be properly satisfied. For such applications, the broadcasting of the message within the destination area is an option, but if the destination area is a long segment, the disadvantage of severe flooding occurs.

A possible improvement is caching common information on nodes, because the query will be resolved faster, and the broadcast area will be limited.



3 = position of vehicle III, as known by vehicle I

Vehicle I unicasts to III, but III knows it is in front of II, therefore II will not receive the message

Figure 8 - Problem with unicast

When a reply to a query is generated, it must first be routed back towards the source, using unicast. When it reaches the approximate source location, one-hop broadcast is used (each node broadcasts the same message only once), hoping that the message will eventually reach the source.

4.1.2 CITY SCENARIO

For city scenarios, the street topology is complex and must be taken into consideration. Simple geographical routing, which could work in highway scenarios because highways are pretty straight, will not work in city scenarios, because the idea of “closer” is not that obvious. If only Cartesian coordinates are considered, the danger of dead-ends appears. A possible solution is to pre-calculate a trajectory, represented as a sequence of streets, and then try to route the message along that trajectory. The trajectory should be calculated by using information about the roads. A trajectory is good if the probability of disconnections due to absence of vehicles along it is very small, therefore roads with more lanes, or roads on which traffic values are known to be high, will be used when computing the trajectory. The trajectory will be placed in the message header, and every node will take it into consideration when choosing the next hop. The concept of “trajectory-based” forwarding was first presented in [14]. The idea of using it for vehicular networks in city scenarios was analyzed in [15], where the position-based routing approach was proven feasible. The algorithm was called “Geographic Source Routing” (GSR). However, the simulations described in [15] involved a region of 6.25km x 3.45km from the city of Berlin and a total number of 955 vehicles. Using our simulation environment (see Section 2) we have simulated more crowded city scenarios, which we believe are closer to the reality. In fact, the authors of [15] were unable to perform simulations with the vehicles’ wireless range set at 250 meters, because they reported disconnections were extremely frequent. Furthermore, they did not even consider the penetration ratio, they considered all vehicles to be equipped with the system, and even then the 250 meters wireless range would not offer acceptable connectivity. Therefore, they set the wireless range for their simulations at 500 meters. We consider that most central regions in cities, especially during rush hour, have much higher traffic densities, therefore we chose to perform our own simulations, using a trajectory-based routing approach.

First, we wanted to check the “obsolete neighbors” problem, which we have described above, (see Section 4.1.1) for the city scenario. We simulated a traffic scenario in a rectangular region (1km x 1km) in downtown Manhattan. We used real TIGER [5] maps of New York. The total number of simulated vehicles exceeds 2000. Screenshots of the simulation are presented in Figure 9. The equipped vehicles exchange the periodic beacons, as previously described. The beacons period was set to 1 second. At random moments, the neighbors list of each equipped vehicle was checked against the real situation. The results are presented in Figure 10.

By comparing the results with those obtained for the highway scenario (Section 4.1.1), it is clear that the number of obsolete neighbors (neighbors which are in fact further than 200 meters, and therefore out of the wireless range) is lower. The explanation is surely the lower mobility of nodes in a city scenario, where speeds are considerably lower than on highways. Furthermore, no clear influence of the penetration ratio can be observed.

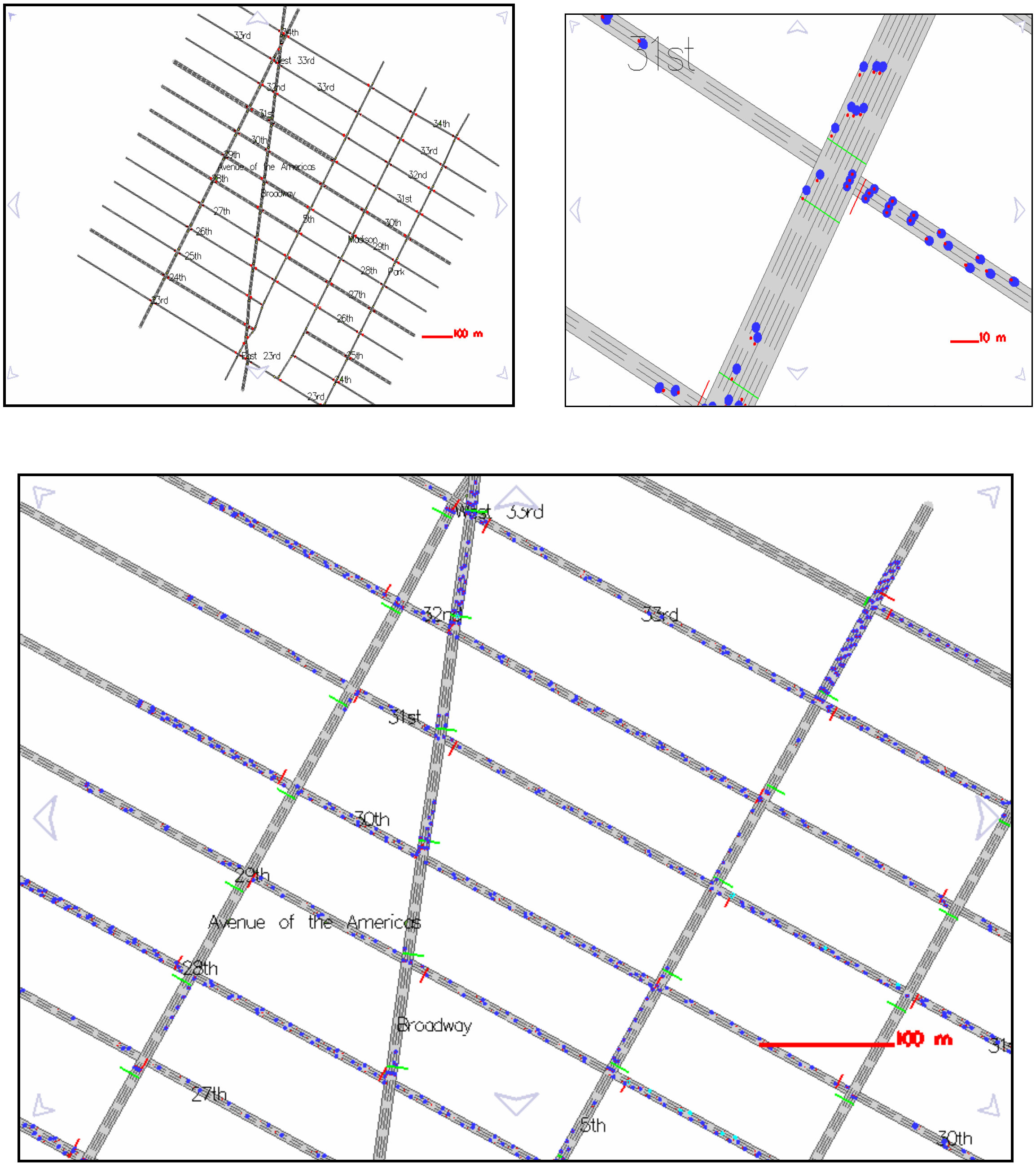


Figure 9 – Screenshots taken during the simulation

EXPIRATION PERIOD = 3 seconds

DISTANCE FROM NEIGHBOR	NUMBER OF SITUATIONS (PENETRATION RATIO=60%)		NUMBER OF SITUATIONS (PENETRATION RATIO=30%)	
(meters)	(count)	(%)	(count)	(%)
<50	207437	**17.21**	73847	**15.66**
50-100	169842	**14.09**	60618	**12.86**
100-150	246624	**20.46**	127700	**27.08**
150-200	575904	**47.78**	207673	**44.04**
200-250	5412	**0.45**	1664	**0.35**
>250	0	**0.00**	0	**0.00**

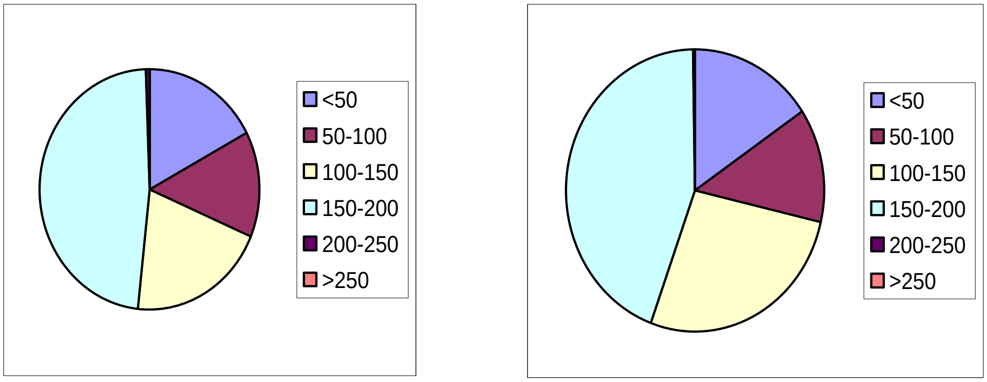


Figure 10 – Accuracy of neighbor information – city scenario

4.2. "POST" MESSAGES

VITP "POST" messages can be used to signal an accident, a traffic jam, or any other interesting/important piece of information (for instance, a gas-station could use "POST" messages in order to announce its presence). Each "POST" message is associated with a destination region. It can be generated directly within it (for example, to warn all vehicles around a node that that node has had an accident), or outside of it. If it generated outside the destination region, then it must be routed based on unicast until reaching it; after it is inside the destination region, every node will broadcast it periodically. A node will stop broadcasting the message when it is no longer in its destination region. The message can also be discarded when a time-out expires.

A mechanism should be designed to avoid too many different messages signaling the same incident to be broadcasted independently inside a region (for example, when an accident occurs, many different vehicles could separately initiate warnings about the same issue). To achieve this, the semantics of the message need to be considered. Such a mechanism would be implemented at the application layer and is outside the scope of our document.

5. IMPLEMENTATION DETAILS

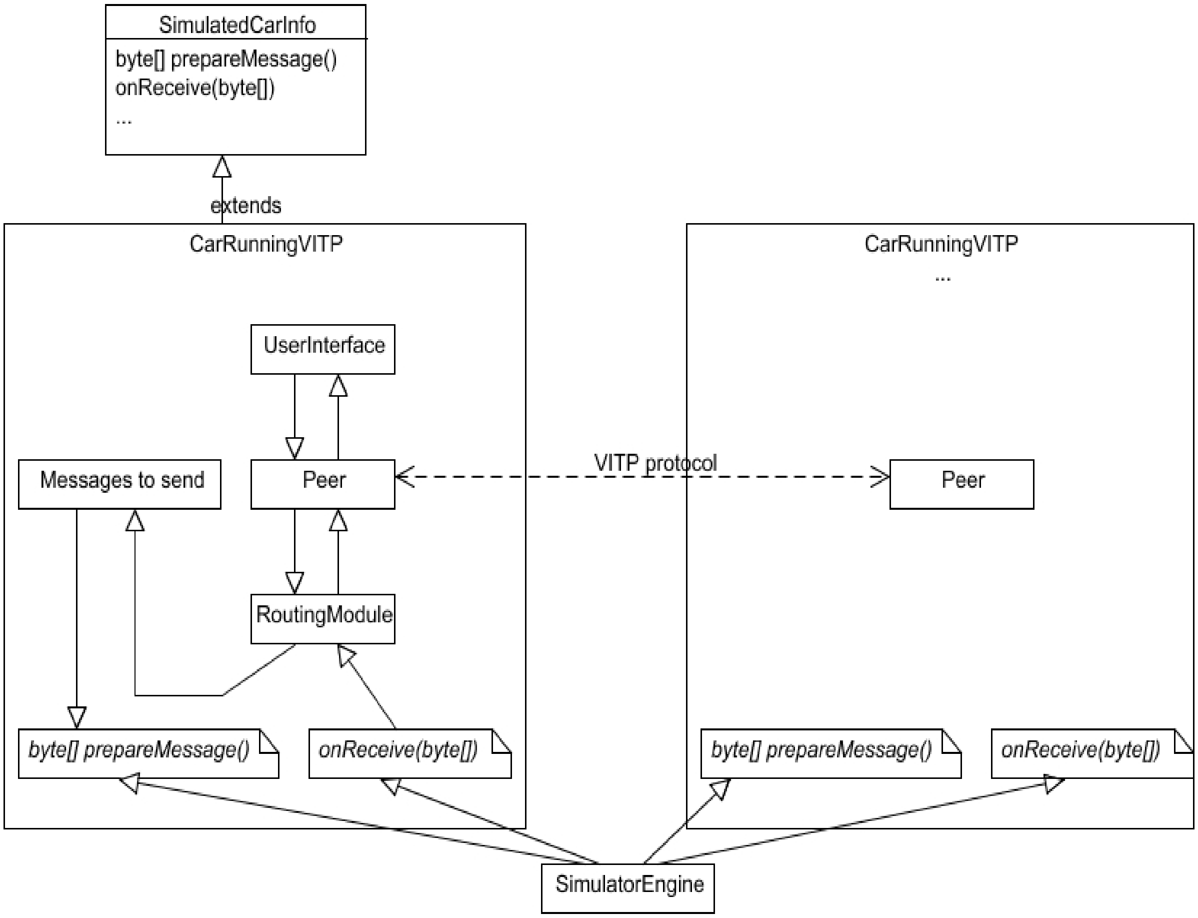


Figure 11 – System architecture

The architecture of the system is presented in Figure 11. The design is modular, multi-layered, and therefore it is scalable and easily extensible.

The simulator engine keeps track of all the nodes and takes care of delivering messages among them. The simulator is a single-threaded, discrete, event-based simulator. The nodes are modeled by the class SimulatedCarInfo, which has 2 important methods: “onReceive” and “prepareMessage”. “onReceive” is called by the simulator engine, when the node has received a message. “prepareMessage” is called by the simulator engine, when the node intends to send a message. Messages, at this level, are seen as simple byte sequences.

The VITP and routing protocols were implemented in the CarRunningVITP class, which extends SimulatedCarInfo and overrides the 2 methods. A CarRunningVITP instance contains a user interface module (actually, this is just a stub), a peer module, and a routing module. Different types of messages are exchanged between the various modules. The peer is the component which implements the VITP protocol, while the RoutingModule implements routing details.

There are 2 distinct message flows. One of them starts when the peer intends to send a message (possibly as a request by a human user or a software component). For instance, if the human user wants to issue a query, a method of the peer will eventually be called. The peer will eventually pass the message to the routing module. Messages sent by the peer contain application-level information, intended to be understood by the peer modules on other nodes. The routing module encapsulates the VITP messages in routing packets and prepares them for sending. When the simulator engine calls the "prepareMessage()" method, the corresponding sequence of bytes is returned, and the simulator engine will take care of sending it, according to the wireless transmission simulator module. The other flow starts when the simulator engine delivers a message to a node. Thus, it calls the "onMessage" method, passing it the corresponding sequence of bytes. It is then decoded by the routing module, and the VITP message is sent to the peer, which deals with it accordingly. For example, it could append useful information to the message, if it's a request, or it could display it to the user in a proper format if it's a POST warning.

The classes which model the different types of messages exchanged between the various components are presented in the UML class diagram in Figure 12.

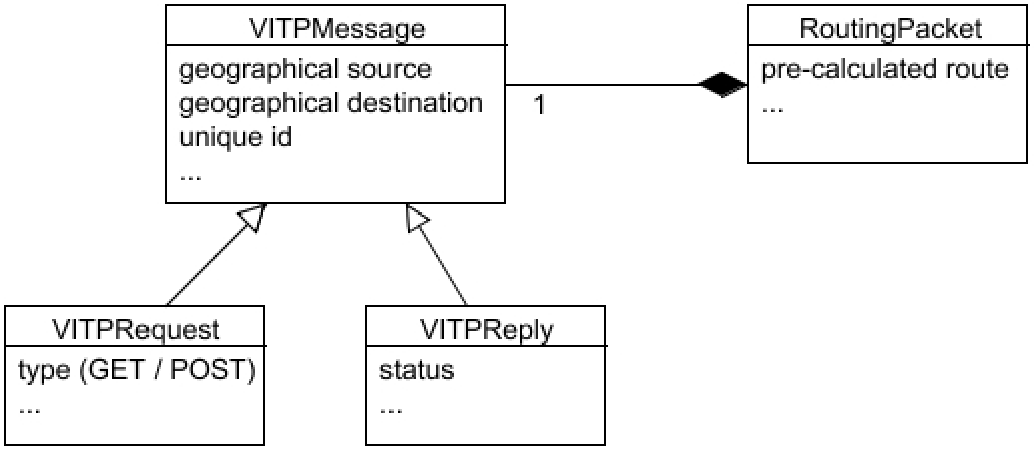


Figure 12 – UML class diagram - Messages

VITP messages are created and understood by the “peer” modules. They contain a geographical source and destination, and can be one of two types: requests and replies. They also contain additional VITP-level information, for instance, the number of nodes a request must visit in the destination area. VITP messages are encapsulated by the routing modules in “routing packets”, which contain additional routing information.

6. SIMULATION RESULTS

6.1 HIGHWAY SCENARIO

The first simulations we have performed were on a typical highway. We used real maps from New Jersey, freely available from [5]. The map represents a section from the NJ Turnpike highway, about 30 kilometers long, together with adjacent roads. We mention once again that the TIGER maps lack information like the number of lanes, or traffic control systems. They also lack altitude information, as they only represent a 2D model of reality. Therefore, there is no way of telling the difference between overpasses and intersections. Thus, it was our job to add some of the extra information, based on common sense rules. There are no traffic lights on the Turnpike, and we have not allowed left turns, which, in reality, are of course made impossible by the physical separation of the opposing directions.

On this highway, we first simulated a typical crowded, but fluent traffic scenario. The scenario involved an average flow of 1400 vehicles per hour per lane, in both directions, on the Turnpike, together with small flows of vehicles seeking to enter the Turnpike from some of the adjacent roads. That has resulted in a crowded highway, but definitely not jammed. For this simulation, we chose to set the number of lanes per direction to 3. In reality, the Turnpike has 6 lanes per direction through most of New Jersey, but there are also long sections with 3 or 4 lanes per direction.

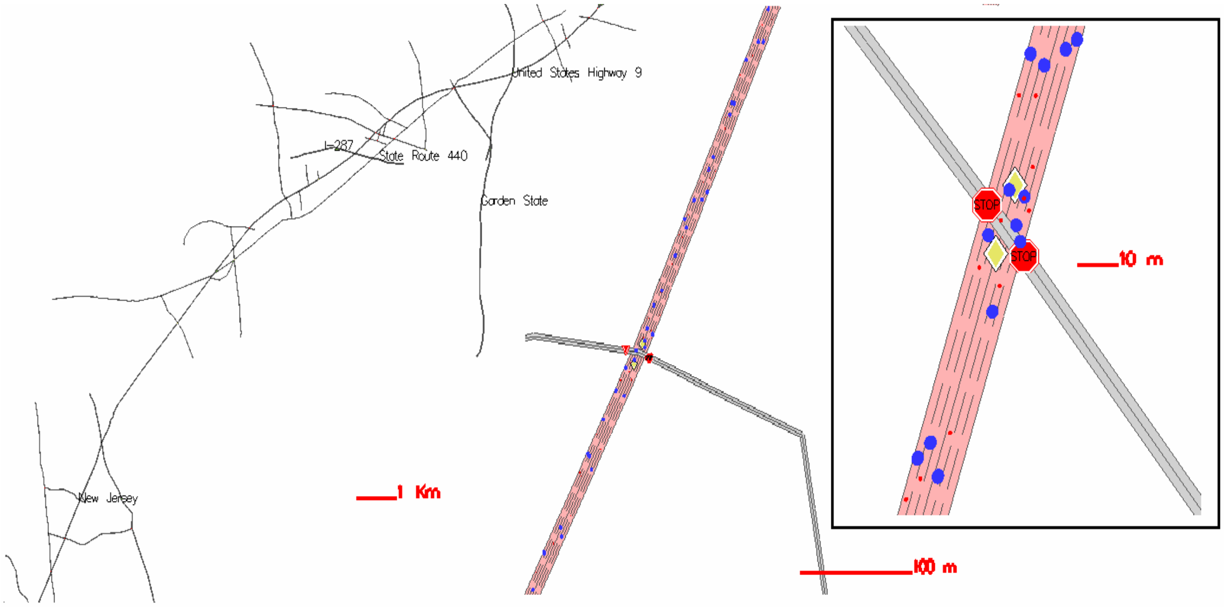


Figure 13 - Screenshots taken during the simulation

6.1.1 QUERY-REPLY SIMULATIONS

On the described scenario, one vehicle was randomly selected every 10 seconds to issue a query message, which was supposed to reach at least N vehicles in a remote destination area. The query and the reply are routed according to the algorithm described in 4.1.

Because it is obvious that the deployment of real-world vehicular networks cannot be done immediately on all vehicles, we consider the penetration ratio to be a very important parameter. It is very useful to develop applications which can work reasonably even with medium or low penetration rates. Therefore, we studied the influence of the penetration ratio on the query-reply application. Equipped vehicles run the described protocols, while unequipped vehicles take no part in the communication: they do not broadcast their position, they do not receive any information and they do not inject any queries. Figure 14 presents results of the simulation. For this simulation, N was set to 5, and a total number of around 200 queries were issued for each scenario. The percent of satisfied queries refers to the number of queries which obtain a reply with some valid information from the destination area. It is worth noting that there are also replies which do arrive back to the source, but which do not come from the destination area, because the original query could not be routed all the way. Instead, they just report the position where the disconnection was met. Such replies are not considered "satisfied" and therefore do not count to the figures presented in Figure 14.

We also studied the influence which the distance between the source of the query and the destination region has on the percent of satisfied queries. Thus, 3 different types of queries were issued, one for a region situated 6 kilometers away, one for 10 kilometers, and one for 18 kilometers.

EQUIPPED VEHICLES	SATISFIED QUERIES DISTANCE = 6.2 KM	SATISFIED QUERIES DISTANCE = 10.2 KM	SATISFIED QUERIES DISTANCE = 18.7 KM
(%)	(%)	(%)	(%)
20	14	1	0
25	33	7	0
30	61	49	11
35	80	67	51
40	92	84	83
45	89	95	93
50	100	97	98
55	100	94	97

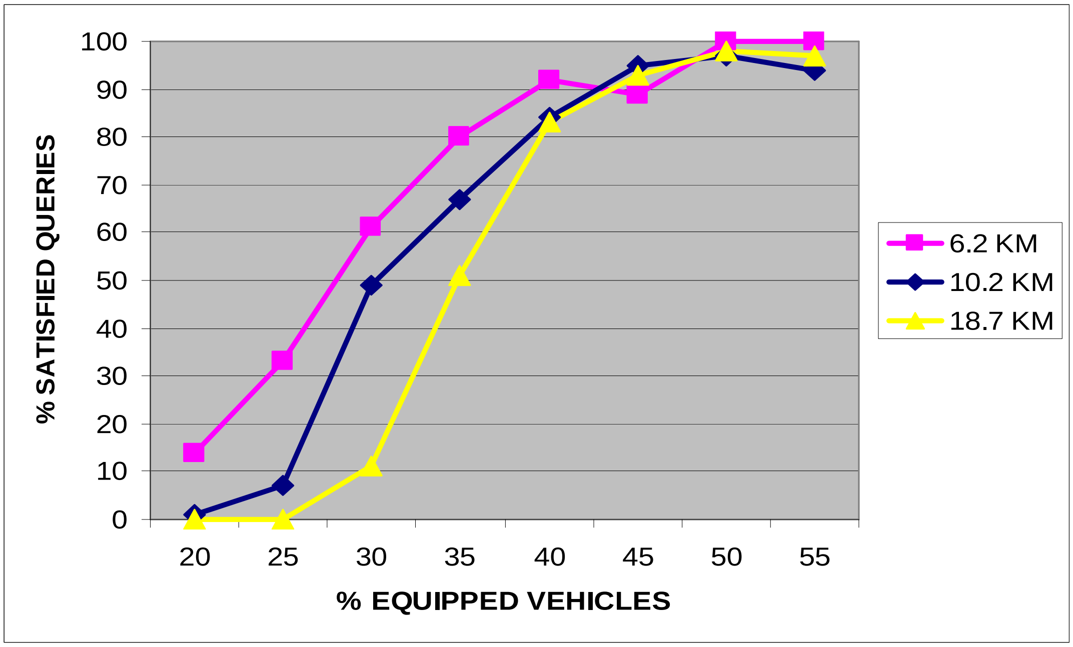


Figure 14 – Simulation results

The graph shows sharp increases in the number of satisfied queries around the area between 25 and 40 percent equipped vehicles. This behavior is in fact not surprising and we are going to describe a simplified model in order to explain it.

Consider a 6-lane highway with a flow of FLOW vehicles per hour on each lane. Let's make the simplifying assumption that all vehicles move at the same speed and the gap between each 2 consecutive vehicles on a specific lane is the same (see Figure 15).

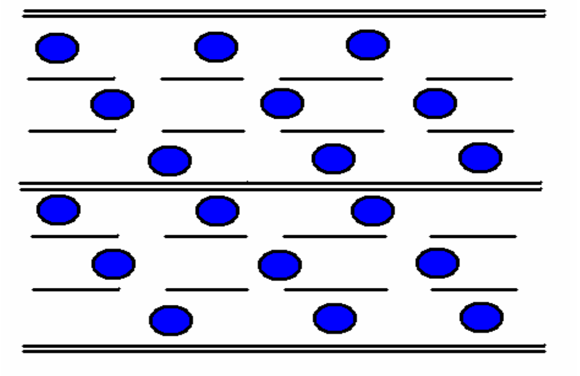


Figure 15 - Simplified, unrealistic model

The gap, in meters, between 2 consecutive vehicles on one lane is given by the simple formula:

GAP = 1000 * SPEED [KM/H] / FLOW [VEHICLES/HOUR/LANE]

The average distance between every 2 vehicles (this time not considering lanes) is therefore:

DISTANCE = GAP / NUMBER_OF_LANES

If we consider that only EQUIPPED percent of the total number of vehicles are equipped with devices capable of running the suggested protocols, it follows that the average distance between every 2 consecutive equipped vehicles is:

DISTANCE_EQUIPPED = DISTANCE * 100 / EQUIPPED

The table in Figure 16 presents the value of this distance, for several different values of the parameters it depends on.

FLOW	SPEED	NUMBER	AVERAGE	EQUIPPED	AVERAGE DISTANCE
PER LANE		OF LANES	DISTANCE	VEHICLES	BETWEEN
			BETWEEN VEHICLES		EQUIPPED VEHICLES
(PER HOUR)	(KM / H)		(METERS)	(%)	(METERS)
1000.00	90.00	6	15.00	**5.00**	**300.00**
1000.00	90.00	6	15.00	**10.00**	**150.00**
1000.00	90.00	6	15.00	15.00	100.00
1000.00	100.00	6	16.67	**5.00**	**333.33**
1000.00	100.00	6	16.67	**10.00**	**166.67**
1000.00	100.00	6	16.67	15.00	111.11
1500.00	90.00	6	10.00	**5.00**	**200.00**
1500.00	90.00	6	10.00	**10.00**	**100.00**
1500.00	90.00	6	10.00	15.00	66.67
1500.00	100.00	6	11.11	**5.00**	**222.22**
1500.00	100.00	6	11.11	**10.00**	**111.11**
1500.00	100.00	6	11.11	15.00	74.07

Figure 16

It is clear that, if the average distance between equipped vehicles is lower than the wireless range, then a routing protocol based on geographical routing has a good chance of succeeding. If, on the other hand, equipped vehicles have an average distance between them greater than the wireless range, then the routing protocol will not work. That leads to the conclusion that there is a "threshold" for the percentage of equipped vehicles above which the protocol will start performing quite well, but below which the protocol has practically no chance of succeeding. That threshold is, obviously, the exact value of the percentage of equipped vehicles for which the average distance between equipped vehicles is equal to the wireless range. However, if we consider the wireless range to be 200 meters (as we have done in the simulation), the threshold, as it results from the table in Figure 13, is somewhere between 5% and 10% equipped vehicles, for the over-simplified scenario presented. Our simulation results (see Figure 14) do show a sharp increase in the number of satisfied queries, but the threshold where this increase occurs is situated between 25% and 40%. It is clear that this difference is caused by the assumptions made in the simplified model, that all vehicles have the same speed, and that the gap between any 2 consecutive vehicles is a constant. These assumptions are clearly not consistent with the movement of vehicles in reality, where more complex phenomena, like platoon formation, occur. Therefore, a more accurate driver behavior model (like the one implemented by our simulator), is bound to provide different results. Having explained the shape of the graphs, it is worth noting that the actual figures are quite promising. Even for quite a big distance between the node originating the query and the destination region (over 18 kilometers), a penetration ratio of 35% equipped vehicles is enough for a value of 50% of satisfied queries.

From the same simulations previously described, we also consider 2 other indicators as being important: the time period passing from the moment when the query is injected until the reply is received, and the number of intermediate nodes through which a message passes, on its way towards the destination and backwards.

Therefore, we have calculated averages for these indicators and the results are shown in Figure 17. The first graph shows the average delays measured between the moment when a query is injected and the moment when the corresponding reply is received. The second graph shows the average number of intermediate nodes through which a message has passed. It must be noted that only satisfied queries (as previously defined) were taken into account for both of these graphs. As expected, the delay, as well as the number of hops, is clearly proportional with the distance between the source of the query and the destination region. The figures themselves are, again, promising, delays smaller than 10 seconds defining a quite responsive system.

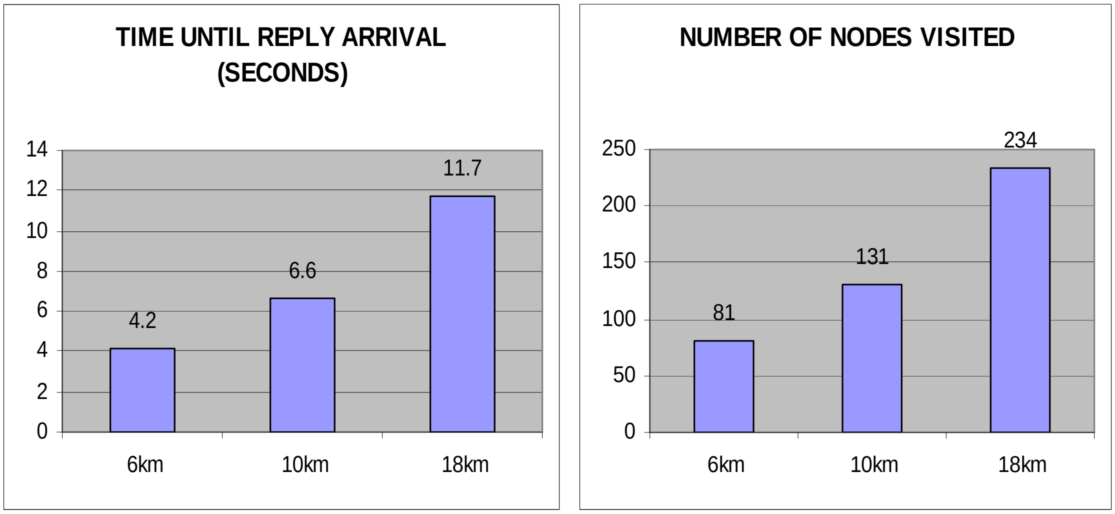


Figure 17

6.1.2 "POST" MESSAGES SIMULATIONS

On the same highway traffic scenario previously described, we wanted to evaluate the performance of the broadcast mechanism used by the routing protocol for "POST" messages.

Therefore, we chose a region of a specified length, L, inside which we injected a "POST" message. How this message is routed towards the region is not relevant to the results presented in this section. It could have been routed by the geographical routing algorithm, or it could have been directly injected from a node already inside the destination region.

In any way, the message is going to be broadcasted by every node periodically, as long as the nodes remain within the destination region; whenever a node leaves the region, it will stop broadcasting the message. Let T be the broadcast period. L, T, and the penetration ratio are the main parameters which we vary, in order to study the performance of the protocol.

First, we wanted to check how the broadcast period affects performance. The period cannot be too small, because a broadcast storm would appear. However, if it is too large, then there is the risk of nodes not receiving the message. We set the length of the destination region to 900 meters and the penetration ratio to 60% and we performed simulations for different values of the broadcast period: 10 seconds, 15 seconds, and 20 seconds.

The graph in figure 18 shows the evolution in time of the percentage of informed vehicles (vehicles which have already received a copy of the message) within the destination region. Naturally, only equipped vehicles are taken into consideration towards these numbers. Periodical increases of this percentage occur, as expected. The period of these increases is equal to the broadcast period. It is clear that, even with a 20 seconds-long broadcast period, the message is not lost. It is kept alive within the destination region and the probability that a vehicle passes through the whole region without receiving the message is practically null.

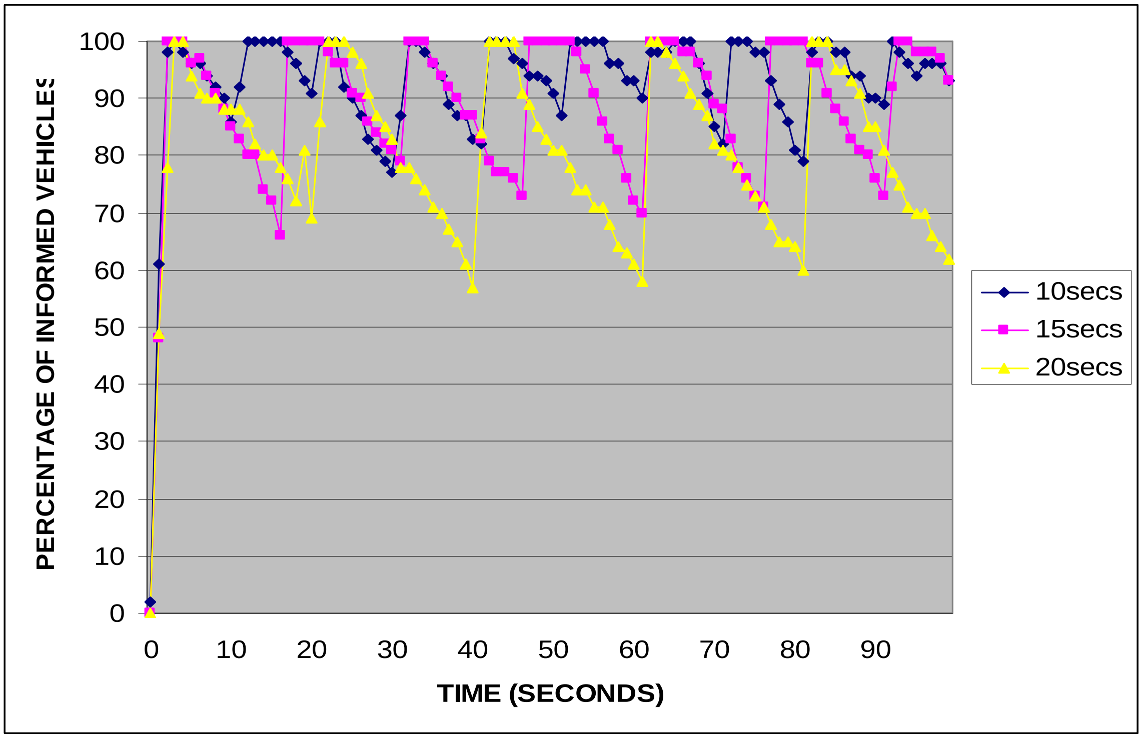


Figure 18

BROADCAST PERIOD	NUMBER OF BROADCASTS
10 sec	1138
15 sec	775
20 sec	548

Figure 19 - Total number of broadcasts within destination region during 200 seconds

Next, we wanted to check the influence of the penetration ratio on the protocol's performance. We set the length of the destination segment to 900 meters, the broadcast period to 10 seconds and we performed simulations with different values for the penetration ratio.

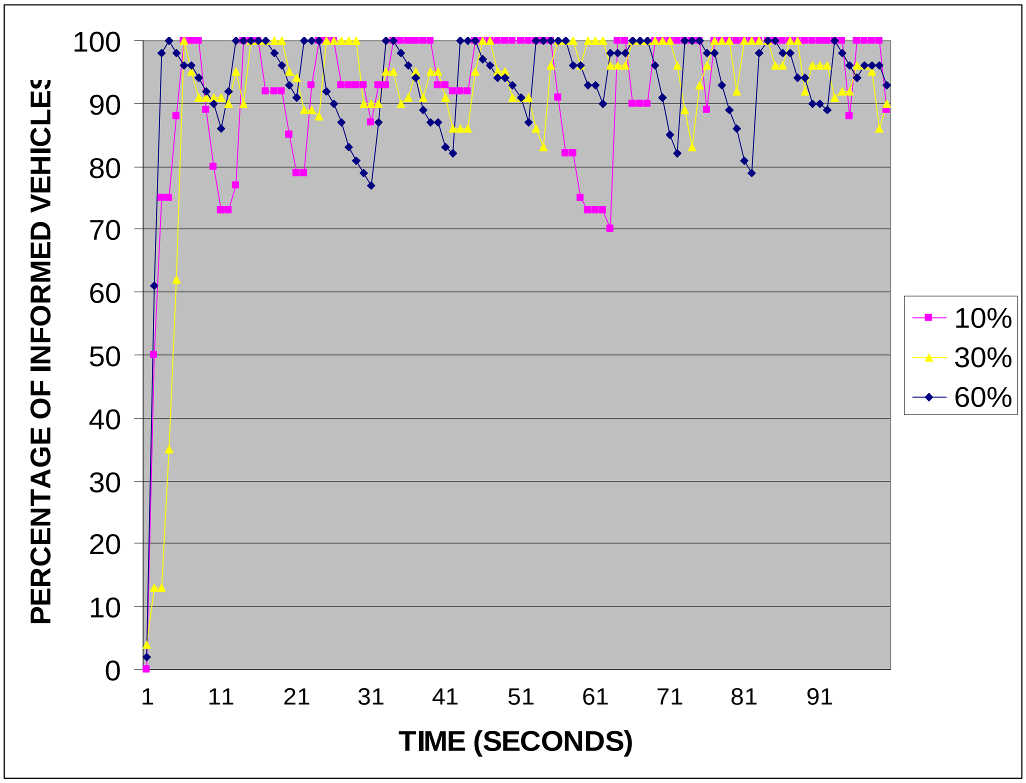


Figure 20

PENETRATION RATIO	AVERAGE NUMBER OF VEHICLES INSIDE THE DESTINATION AREA
10%	8
30%	26
60%	58

Figure 21

The graph in Figure 20 shows the evolution of the percentage of informed vehicles, for three different penetration ratios: 10%, 30% and 60%. The periodical increases of this percentage can be clearly seen on the curve corresponding to a 60% penetration ratio. On the other hand, for 10% and 30% penetration ratio, this periodicity is no longer clear. The explanation is the lower number of equipped vehicles in the destination area, which makes the computing of a percentage less significant.

However, even for a very small penetration ratio (10%), it is clear that the 10-second broadcast period is sufficient. The message is kept alive within the destination area and practically all equipped vehicles receive it. It must be noted that the penetration ratio required for a proper functioning of the “POST” messages broadcast mechanism is lower than that required by the query-reply protocol, which was proven not to work for levels of the penetration ratio under 20%, for the same traffic scenario (see Section 6.1.1). Finally, we wanted to check the influence of the length (L) of the destination area. The graph in Figure 22 shows the evolution of the percentage of informed vehicles, for a penetration ratio set at 60%, a broadcast period of 10 seconds, and 3 different values of L: 500 meters, 900 meters and 1400 meters. From the data, we conclude that no significant influence can be observed and the algorithm performs well even for large regions.

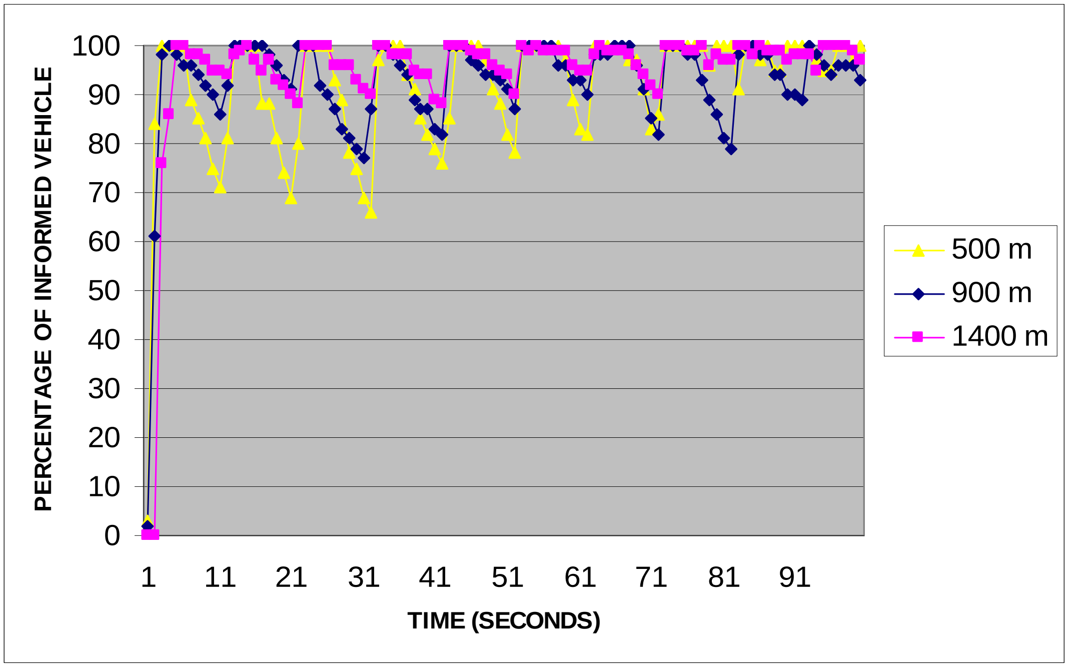


Figure 22

REGION LENGTH	NUMBER OF BROADCASTS
500 meters	703
900 meters	1138
1400 meters	2018

Figure 23 - Total number of broadcasts within destination region during 200 seconds

6.1.3 BANDWIDTH REQUIREMENTS

The protocols we have studied are not dependent of the MAC layer protocol. However, no matter what MAC layer protocol is going to be used, bandwidth will be an important issue. The question arises, whether the higher-level protocols' bandwidth requirements are realistic, and can be satisfied by the MAC layer. Therefore, we try to determine the approximate bandwidth requirements.

We consider again the simplified model presented in Section 6.1.1 (see Figure 15). Let’s consider a highway region of 200 meters. The number of equipped vehicles within this region is given by:

N_EQUIPPED = 200 * FLOW * N_LANES * PERCENT_EQUIPPED /
(1000*SPEED*100)

For an explanation of this formula and the notations, see the equations in Section 6.1.1. Let’s assume that the beacons’ broadcast period for each equipped vehicle is 1 second. A possible estimation of the beacon’s size is given in Figure 24.

FIELD	DETAILED DESCRIPTION	SIZE (BYTES)
NODE POSITION	ROAD INDEX	4
	POINT INDEX	4
	LANE_NUMBER	1
	DIRECTION	1
NODE SPEED		2
	TOTAL SIZE:	**12**

Figure 24

It is clear that a beacon’s size is not large. It seems safe to consider a 20 bytes size for them. The total necessary bandwidth, if we only consider vehicles within the 200 meters region, is:

BANDWIDTH [bps] = N_EQUIPPED * BEACON_SIZE [bytes] * 8

The table in Figure 25 contains the estimation of this bandwidth, for several values of some of the parameters it depends on.

FLOW	NUMBER	SPEED	EQUIPPED	BEACON	BANDWIDTH
	OF LANES		VEHICLES	SIZE	
(vehicles/hour/lane)		(km/h)	(%)	(bytes)	(Mbps)
1000	6	80	50	15	**0.88**
1000	6	80	30	15	**0.53**
1000	6	100	50	15	**0.70**
1500	6	80	50	15	**1.32**
1500	6	80	30	15	**0.79**
1500	6	100	50	15	**1.05**
2000	6	80	50	15	**1.76**
2000	6	80	50	20	**2.34**
2000	6	60	30	15	**1.41**
2000	6	60	30	20	**1.88**

Figure 25 - Bandwidth estimation

The table in figure 25 refers only to the bandwidth required by the periodic beacons sent by all the equipped vehicles. Furthermore, the needed bandwidth is in fact higher because vehicles within the 200 meters considered region also share the medium with adjacent vehicles, just outside of the considered region.

On top of this, bandwidth is also required for the routing protocols we have studied. While the unicast-based routing of queries or replies requires only a very small amount of bandwidth, the most consuming part is the broadcast mechanism used by “POST” messages, which must be kept alive within a region. The tables in Figures 19 and 23 suggest values of over 1000 broadcasts in a 1000 meters long region, during 200 seconds. That means that, in a region of 200 meters, the approximate number of broadcasts per second is 1. The required bandwidth is significantly lower than that required by the periodic beacons, but we must take into account the fact that multiple “POST” messages could well be simultaneously alive within the same region, and that will lead to increased bandwidth requirements.

Existing MAC layer protocols, like 802.11a, 802.11b and 802.11g are candidates to be used for vehicular networks. The promised bandwidth clearly exceeds the approximate required values we have presented above. However, the real performance of the 802.11 variants in highly-mobile ad-hoc networks has not been sufficiently studied.

Some results [16] indicate that real performance when testing currently available 802.11 technologies in vehicular environments is very poor. Frequent disconnections appear, the wireless range and the bandwidth are lower and delays are higher than expected.

Dedicated Short Range Communications (DSRC) is a block of spectrum in the 5.850 to 5.925 GHz band allocated by US Federal Communications Commission to enhance the safety and the productivity of the transportation system. It was especially allocated for vehicle communication; however work on standardization has been very slow. 802.11p is an IEEE standard especially targeting vehicular computing applications. Its goals are higher range (up to 1 km), and good performance in high-speed environments (up to 200km/h). However, this standard is still under development and progress has been very slow recently.

We must conclude that the necessary lower layer technology for vehicular computing applications does not exist at the present moment, but work is done in that area, and improvements can be expected in the future.

6.2 CITY SCENARIO

We simulated a traffic scenario in a rectangular region (1km x 1km) in downtown Manhattan. We used real TIGER [5] maps of New York. We also set the “one-way” and “number of lanes” characteristics for the streets, according to the reality. The total number of simulated vehicles exceeds 2000. Several independent flows of vehicles were simulated simultaneously, the result being a crowded scenario, with vehicles distributed on most streets, with queuing at traffic lights. A “warm-up” time period of 200 seconds was introduced before making any measurements, in order to allow the vehicles to form a realistic scenario. Screenshots of the simulation are presented in Figure 9.

In order to validate our scenario, we computed the average speed of all vehicles within the region. The evolution of the average speed, obtained after running the scenario

3 times, is represented in Figure 26. We have calculated the total average vehicle speed and obtained 6.4 miles per hour, or 10.3 kilometers per hour. According to some official studies [17] [18], the average vehicle speed in downtown Manhattan is between 4 miles per hour and 8 miles per hour. Therefore, we conclude that our scenario is an accurate representation of a real traffic scenario in downtown Manhattan.

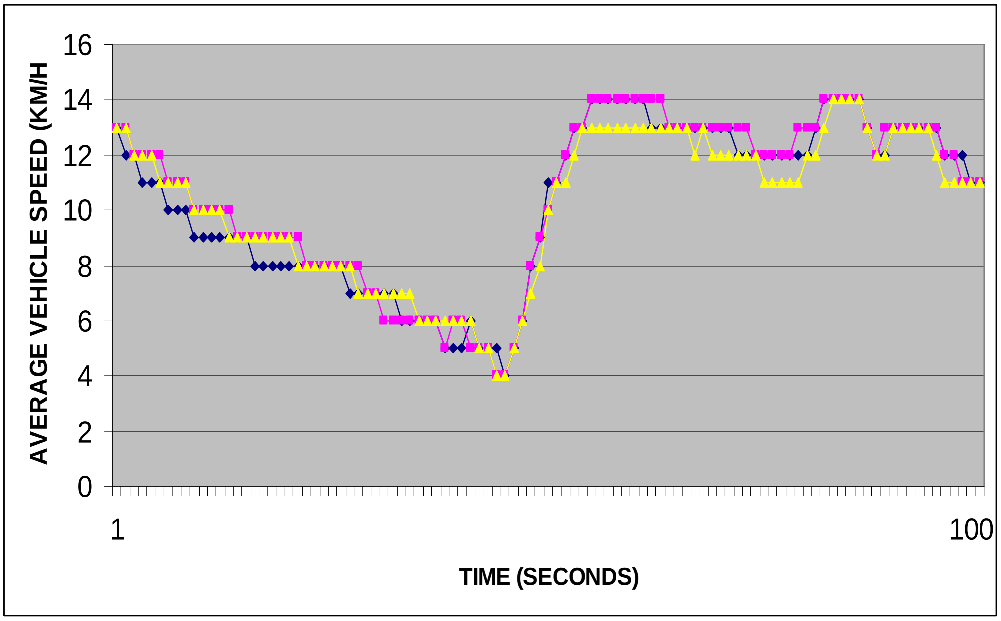


Figure 26 - Average vehicle speed in Manhattan scenario

6.2.1 "POST" MESSAGES SIMULATIONS

The equipped vehicles in the simulated scenario exchange periodic beacons, with a 1-second time period. We wanted to check the dissemination of a VITP "POST" message inside the whole region. Thus, we injected a VITP "POST" message on a node within the region, and then we periodically counted the "informed" vehicles, the vehicles which have already received a copy of the message. We set the broadcast period (the period with which each informed node rebroadcasts the message) to 30 seconds and checked the influence of the penetration ratio on the dissemination process. The results are represented in the graph in Figure 27.

Even with a low, 20% penetration rate, the message is still kept alive within the whole region, among the equipped vehicles, with a 30 seconds broadcast period. It must be noted that the percentage represented in the graphs is computed by taking only equipped vehicles into consideration. This is natural, as the unequipped vehicles do not take any part in the communication process.

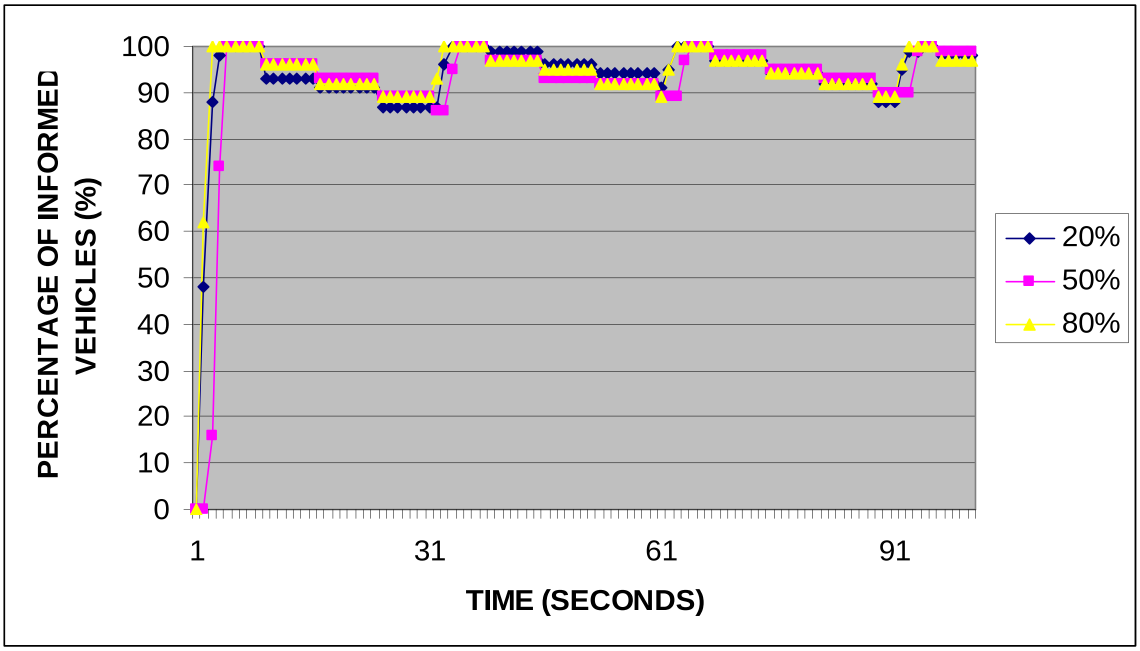


Figure 27

Next, the influence of the broadcast period was checked. The broadcast period is the period with which every informed vehicle rebroadcasts the message. We set the penetration ratio to 20% and varied the broadcast period. The results are presented in the graph in Figure 28. They show that, even with a very long broadcast period (of 90 seconds), there is no problem in keeping the message alive inside the region, and, as more and more vehicles enter the region, they too will eventually receive the message. The drawback of having a long broadcast period is that quite a lot of time may pass from the moment when a node enters the region until it receives the message.

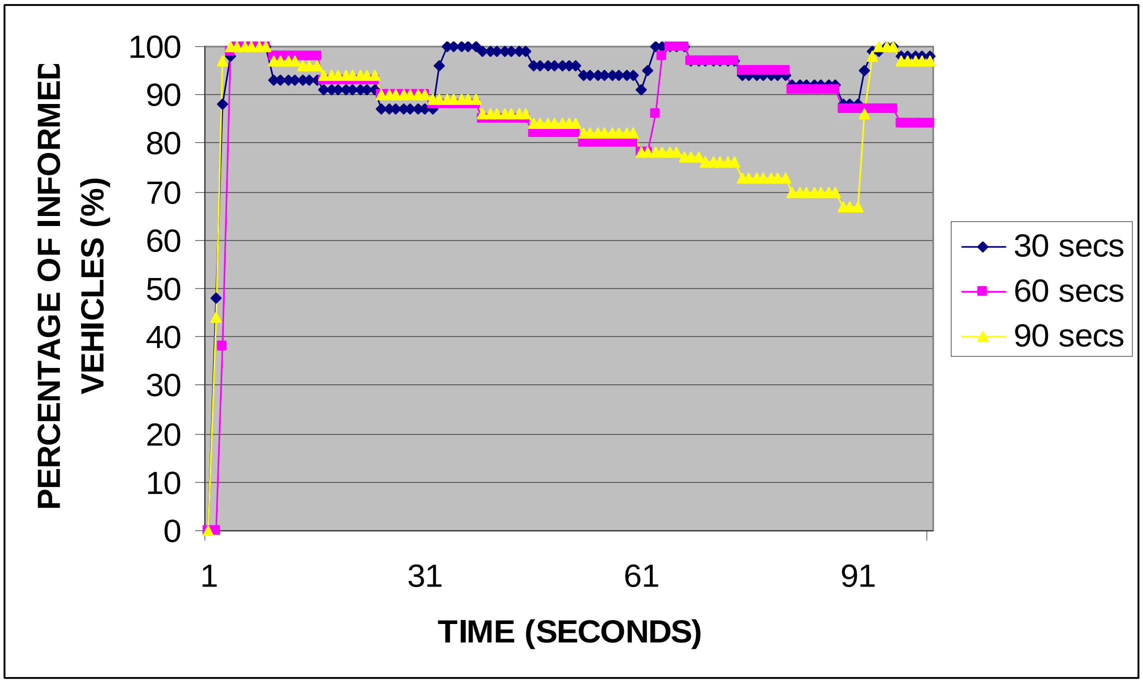


Figure 28

6.2.2 BANDWIDTH REQUIREMENTS

For our traffic scenario, we have calculated the average number of neighbors a node has in its wireless range (set at 200 meters), as a function of the penetration ratio. The results are shown in Figure 29.

While the number of neighbors may seem unusually large, it must be noted that we have not taken into consideration any influence which buildings or other obstacles could have on the wireless transmission. The wireless range is simply a 200 meters circular area, with the node in its center. In reality, the radio signal will not propagate well in areas with tall buildings, therefore the number of neighbors a node will actually be able to communicate with will be smaller. Furthermore, the Manhattan map is particularly crowded, with a very dense network of streets. Other city scenarios could well have a lower density of streets; therefore the average number of neighbors would be further reduced.

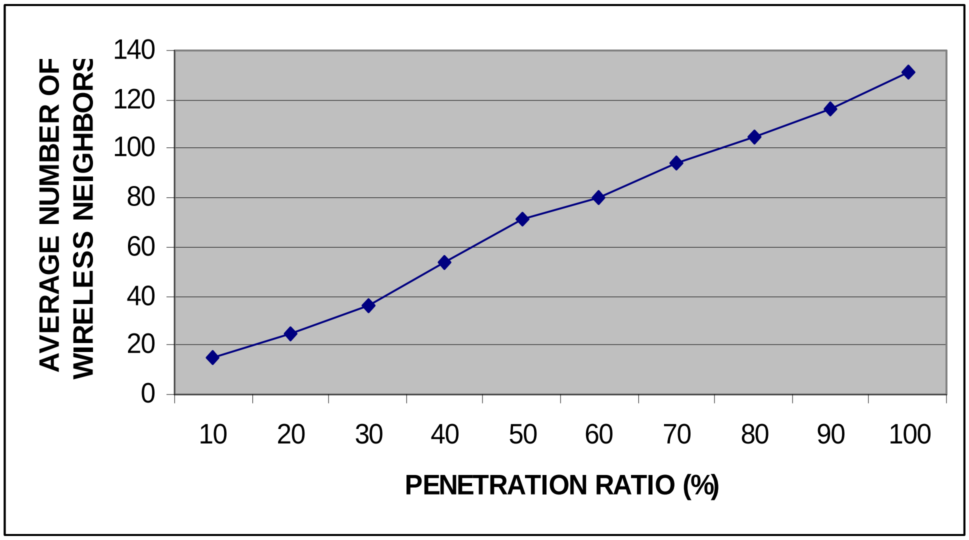


Figure 29

Next, we try to approximate the bandwidth required by the periodic beacons. The table in Figure 30 shows, as expected, higher bandwidth values than the ones obtained for the highway scenario (see Figure 25). We must mention that, in reality, the required bandwidth would be influenced by 2 more factors, which we have not considered. Buildings and other obstacles would limit the wireless range; therefore the required bandwidth would be lower. However, there would also be an influence from vehicles just outside of the wireless range area considered, because they would also compete for the same medium, therefore increasing the required bandwidth.

EQUIPPED VEHICLES	BEACON SIZE	BANDWIDTH
(%)	(bytes)	(Mbps)
40	15	**6.45**
40	20	**8.59**
50	15	**8.20**
50	20	**10.94**
60	15	**9.38**
60	20	**12.50**

Figure 30

We must conclude that the figures in the table in Figure 30 are a very raw approximation of the bandwidth required by the periodic beacons. On top of this, more bandwidth would be required by the routing of queries, as well as by the dissemination of "POST" messages.

However, the results clearly show that, especially in crowded city scenarios, bandwidth is a key issue for vehicular networks. Higher-layer protocols should be developed keeping in mind that bandwidth is extremely limited, and lower-layer protocols and technologies which can offer more bandwidth are needed.

7. EXAMPLES OF APPLICATIONS

The proper routing of messages in a vehicular network opens the door for a large number of applications. Queries initiated by the driver or by a software component could provide vital real-time information about congestions, accidents, special events, information which could otherwise not be available. Drivers would thus be able to properly choose their routes, avoid dangerous zones and minimize the total delay of a journey.

Access to additional information would also be made possible. For instance, a driver could book a room at a hotel, while still tens of kilometers away, without having to know its address beforehand, without first having to park the vehicle, go inside and ask for the prices and so on. We must stress that there is no cost associated with sending messages through short-range wireless connectivity, if unlicensed bandwidth is used (as is the case with 802.11). Thus, the only cost required in order to have access to all the services a VANET can provide is the cost of the wireless device, the computing device and the software running on top of them.

More intelligent traffic control systems could also be imagined. The driver of an emergency vehicle could clear its route well ahead, by commanding the green light for all traffic-lights ahead, or by informing other vehicles that it is approaching.

For vehicular ad hoc networks, security is a big concern and some authors have focused on it [19]. Basically, with no security mechanism, all sorts of problems could occur. For the “query-reply” problem, any node which sees the message could provide false information and then send it further. A driver could deliberately post warnings of accidents or congestion for some roads, thus hoping that other drivers will avoid those regions, and leaving her/him with a congestion-free road. Or a driver could pretend she/he is driving an emergency vehicle and command all other vehicles to stop and all traffic lights to show the green light for her/him. However, in today’s world, a driver could also mount a siren, or blinking red and blue lights on a vehicle, and start using them while in traffic. Of course, she/he would be breaking the law, and the number of drivers willing to risk doing that, just to make a journey several minutes shorter, is practically null. Thus, strict regulations might be a better way to protect against some attacks than purely technical methods.

The design of security mechanisms ultimately involves some sort of identification method, and this raises the privacy problem. Drivers will not accept their actions and journeys to be easily tracked by anyone. They will be afraid that the query asking “what is your speed?” might be issued by a police vehicle and they will want to be sure that their identity is not revealed, if they are driving above the speed limit, or at least that they will not be fined. As a conclusion, we believe legal aspects related to vehicular networks are a very interesting and important field of study.

8. CONCLUSIONS

Vehicular ad-hoc networks can provide numerous benefits. Most of the applications which could be built on top of vehicular networks involve multi-hop data delivery. Our document tried to address some of the difficulties of data forwarding in VANETs. We focused on 2 specific routing challenges: the query-reply issue and the posting of a message. We developed our own integrated simulation tool, comprising a complex, realistic microscopic vehicular traffic simulator, and a wireless transmission simulator. We used our tool to perform simulations for highway traffic scenarios and for city scenarios. We came to the conclusion that the query-reply problem and the posting problem are solvable in a VANET, by using protocols based on geographical routing, but only in sufficiently dense vehicular traffic scenarios. We evaluated bandwidth requirements and came to the conclusion that advances in lower layers technology are still required, for good performance of real vehicular networks.

REFERENCES

[1] A.Festag, H. Fußler, H. Hartenstein H., A. Sarma and R. Schmitz - "FLEETNET : Bringing Car-To-Car Communication into the Real World". In *Proc. of the 11th World Congress on ITS,* Nagoya, Japan, October 2004

[2] M.Dikaiakos, S.Iqbal, T.Nadeem, L.Iftode - VITP: "An Information Transfer Protocol for Vehicular Computing". In *Proc. of the 2nd ACM International Workshop on Vehicular Ad Hoc Networks (VANET)*, September 2005.

[3] D.R.Choffnes and F.E. Bustamante – "An Integrated Mobility and Traffic Model for Vehicular Wireless Networks". In *Proc. of the 2nd ACM International Workshop on Vehicular Ad Hoc Networks (VANET)*, September 2005.

[4] A. Saha, D. Johnson – "Modelling Mobility for Vehicular Ad-hoc Networks" - *ACM International Workshop on Vehicular Ad Hoc Networks (VANET)*, 2004.

[5] US Census Bureau, Topologically Integrated Geographic Encoding and Referencing system (Tiger Line) - http://www.census.gov/geo/www/tiger.

[6] R.Wiedemann – "Simulation des Straßenverkehrsflusses". Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe, Heft 8 (1974)

[7] R.Wiedemann – "Modelling of RTI-Elements on multi-lane roads". In: *Advanced Telematics in Road Transport* edited by the Comission of the European Community, DG XIII, Brussels (1991)

[8] VISSIM - http://www.english.ptv.de/cgi-bin/traffic/traf_vissim.pl

[9] M.Fellendorf, P.Vortisch – “Validation of the Microscopic Traffic Flow Model VISSIM in Different Real-World Situations” (2000)

[10] J. Blum, A. Eskandarian, and L. J. Hoffman – “Challenges of Intervehicle AdHoc Networks” - *IEEE Transaction on Intelligent Transportation Systems*, 5(4):347--351. 2004.

[11] L. Briesemeister, L. Schafers, and G. Hommel - “Disseminating messages among highly mobile hosts based on inter-vehicle communication”. *IEEE Intelligent Vehicles Symp.*, Dearborn, MI, October 2000.

[12] Z.D.Chen, H. Kung and D.Vlah - “Ad Hoc Relay Wireless Networks over Moving Vehicles on Highways”. In *MobiHoc* (2001).

[13] H.Wu, R.Fujimoto, R.Guensler, M.Hunter – “MDDV: A Mobility-Centric Data Dissemination Algorithm for Vehicular Networks”. In *Proc. of the 1st ACM International Workshop on Vehicular Ad Hoc Networks (VANET)*, October 2004.

[14] D. Niculescu and B. Nath – “Trajectory Based Forwarding and Its Applications”. In *MobiCom'03* (2003).

[15] C. Lochert, H. Hartenstein, J. Tian, H. Fußler, D. Herrmann, and M. Mauve – “A Routing Strategy for Vehicular Ad Hoc Networks in City Environments”. In *Proc. of IEEE Intelligent Vehicles Symposium (IV2003),* pages 156–161, Columbus, OH, June 2003.

[16] P. Shankar, M. T. Alam, S. Musharoff, N. Ravi, C. V. Prados, V. Gradinescu, R. Diaconescu, C. Gorgorin and L. Iftode – “Outdoor Experience with the TrafficView Application” (2006)

[17] “Thru streets – An Innovative Approach to Managing Midtown Traffic” – New York City - Department of Transportation – March 2004

[18] “Harlem/Morningside Heights Transportation Study” – New York City – Department of Transportation – September 2005

[19] J.-P. Hubaux, S. Capkun, and J. Luo – “The security and privacy of smart vehicles”. IEEE Security & Privacy Magazine, 2(3), 2004.