Realistic Urban Scenarios Simulation for Ad Hoc Networks

Abdoul-Kader Harouna Souley, Soumaya Cherkaoui
Department of Electrical and Computer Engineering
Université de Sherbrooke (QC), Canada

{Abdoul-Kader.Harouna, Soumaya.Cherkaoui}@USherbrooke.ca

ABSTRACT

Realistic simulation scenarios are critical to correctly assess the performance of mobile ad hoc networks. This paper presents a tool to generate realistic mobility traces for MANET simulations. A new mobility module called AMADEOS was developed as an extension for the CANUMobisim framework [1]. AMADEOS makes it easy and fast to automatically generate realistic mobility. It allows editing spatial environments with polygonal obstacles to be used within simulations. It also allows visualizing an animation of the generated mobility traces. To model mobility for simulation environments, a new mobility model was created that takes into account obstacles. A new propagation model based on ray tracing was also implemented as part of AMADEOS.

Keywords: mobility models, realistic scenarios, Network Simulator-2, ray tracing propagation

1. INTRODUCTION

Nearly all published works on MANETs (Mobile Ad hoc NETworks) use simulations. However, simulation scenarios often make simplified assumptions about the mobile network. This is because most commonly used simulators, such as Network Simulator-2 (NS-2) [23] and Glomosim [24], do not include capabilities to easily create realistic scenarios. We mainly focus here on two important scenario parameters: nodes mobility and signal propagation. The work presented in this paper is aimed at facilitating realistic mobility modeling in MANETs simulations. The AMADEOS (Advanced Mobility Models for AD Hoc NEtwOrk Simulations) module was developed as an extension of the CANUMobisim framework [1], an existing tool for mobility generation. AMADEOS makes easier the usage of CANUMobisim by allowing graphically creating and editing simulation environments and automatically generating corresponding configuration files. It extends the capabilities of CANUMobisim by allowing to model environments that also comprise obstacles. Furthermore, it allows the animation of the generated mobility traces in order to better visualize node movements in the mobility simulation scenarios. A second function of AMADEOS is to model a realistic radio propagation model. Current, existing propagation models in NS-2 and Glomosim do not account for obstacles in simulation environment and therefore do not use advanced signal propagation models. Also in this work, we have implemented a new wireless propagation model based on ray tracing method that is integrated after to NS-2 and Glomosim simulation environments.

This paper is organized as follows. First, we review related works on realistic mobility modeling. Second, we present the extension we developed to facilitate the creation of realistic mobility traces. Third, we describe the new developed mobility model that takes into account obstacles. Finally, we present our new propagation model based on ray tracing.

2. RELATED WORKS ON MOBILITY MODELING

A survey of ad hoc mobility models like the random direction model, the random Gauss-Markov model, and the Brownian walk can be found in [8]. The most widely used mobility model is the random way point mobility. Nevertheless, works such as the one described by Yoon and al [3] state that "random waypoint is considered harmful" because it does not give a uniform distribution of nodes in a simulation environment, which in turn affects the connectivity graph on which depends the assessment of the simulated MANET protocols. In order to make scenarios more realistic, some new mobility models have been proposed in the last five years. Jardos and al. [2] developed the first obstacle mobility model for ad hoc scenarios simulation. In [11], pathways were constructed by using a Voronoi diagram of the vertices of polygonal obstacles. In this last model, the mobility model becomes a simple mobility restricted on the created Voronoi graph. Stepanov and al. [5] proposed the graph-walk mobility model similarly to the random waypoint model with the difference that in their model the movement is restricted on a graph. After the graph-walk model, Stepanov and al. [5] developed the CANUMobisim framework [1], a powerful realistic mobility trace generator.

3. EXTENDING THE CANUMOBISIM FRAMEWORK

CANUMobisim [1] is a mobility generation tool, implemented for the CANU (Communication in Ad-hoc Networks for Ubiquitous Computing) project at Stuttgart's university in Germany. This tool can model many interesting mobility models such as smooth mobility, pedestrian mobility, graph walk mobility, fluid vehicle traffic, activity based mobility, etc... It needs the creation of an XML (Extended Markup Language) configuration file for modeling mobility patterns. The mobility traces can then be generated by CANUMobisim in two formats: NS-2 mobility traces and GloMoSim mobility traces. The best benefit of the CANUMobisim tool is that the mobility pattern is not purely random, but controlled by a transition matrix. This matrix is derived from a Markov graph. A Markov graph is an automaton with probabilistic transitions. Probabilities of outgoing transitions sum up to one for every state as shown in Figure 1. The vertices of the graph represent activity locations that nodes might visit and the edges model the connections between these locations.

Before writing the XML configuration file, one needs to fully understand its edition instructions. This is one of the limitations of CANUMobisim. The number of code lines to be written depends on the number of activity location points and their transition edges. The complexity of the XML file length is a multiple of $O(n+t)$ with n and t being respectively the number of activity locations and the number of transition edges. The mean number of lines needed for a single activity location or transition is four. In Figure 1, the very simple example graph with five location points needs an XML file with more than 60 code lines. Also, CANUMobisim can use some geographic maps. It can import only GDF (Geographic Data File) [11] and AWML (Augmented World Modeling Language) [12] data files. Data files in either format are not easy to find because they have copyright restrictions. This is another limitation of CANUMobisim. The extension we developed for the CANUMobisim tool is aimed on the one hand at overcoming the aforementioned limitations, and on the other at introducing obstacles modeling in the simulated environment. The latter aspect is used both for generating more realistic mobility scenarios, and for modeling signal propagation. In fact, a new propagation model based on ray tracing was implemented within AMADEOS [25]. AMADEOS also allows visualizing the final mobility traces generated by CANUMobisim.

.3.1 AUTOMATING THE XML FILE CREATION

Let consider the graph model of the city center proposed for the graph-based mobility model in [5]. Figure 2 shows the graph with 65 numbered vertices and 85 transitions connections. The XML file length needs at least 600 ((85+65) x4) code lines and its creation would take a lot of time. AMADEOS automates the generation of the XML file. It allows drawing the graph and setting the probability values at each state. The graph shown in Figure 2 was drawn using AMADEOS. By default, the probability is uniformly distributed among all outgoing transitions as shown in the state (S) property dialog in Figure 2. The visual graph can then be automatically exported into the right XML format. The simulation parameters that can be found in this XML file are the terrain dimension, the simulation time, the number of nodes and their velocity, the type of generated traces (NS-2 or Glomosim), the Markov graph, etc….

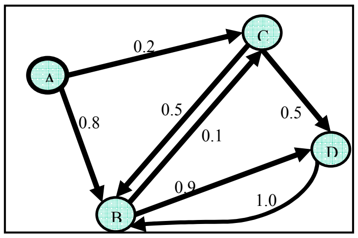
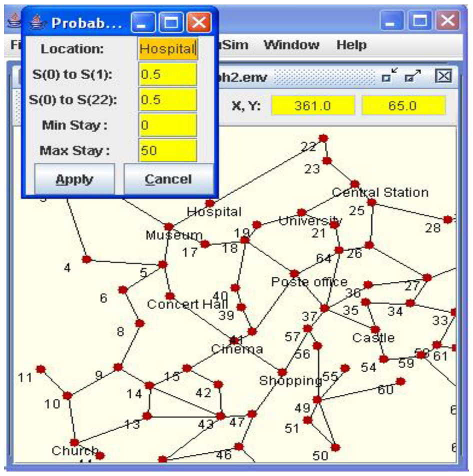


Figure 1: An activity graph



Location: Hospital
S(0) to S(1): 0.5
S(0) to S(22): 0.5
Min Stay : 0
Max Stay : 50
X, Y: 361.0 65.0

Figure 2: A realistic activity graph

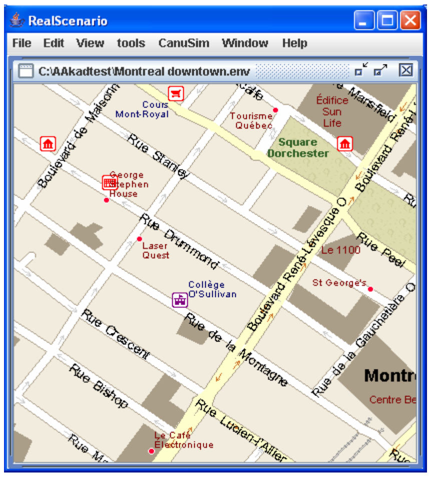
3.2 Introducing obstacles in the environment

AMADEOS is able to create realistic simulation environments with 2D polygonal obstacles. One can edit urban environments by drawing buildings and streets. It is also possible to import/export the footprints of buildings and houses of an urban area to a DXF (Drawing eXchange Format) file. DXF is a well known drawing format developed by Autodesk Inc™ [10]. A user can import real geographic data files previously converted into the DXF format. Alternatively, a user can also load any map as a background image and then redraw all the fundamental objects of the map image. We used this method to create in a few minutes, an urban environment of a landscape in downtown Montreal (Canada). Figure 3 and Figure 4 show respectively the image map and the corresponding environment drawn with the user interface of the AMADEOS.

3.3 Visualizing the mobility traces

After creating the XML file, one can generate the mobility traces by using the CANUMobisim framework and then visualize the traces with AMADEOS. All the objects (building, roads, etc) present in the simulation environment are shown. All details of nodes movement during simulation are also shown. This is very useful as one can return and change the environment parameters if not satisfied with the mobility pattern. Figure 5 shows the automatic visualisation of the Montreal map of Figure 3 and Figure 4 in a scenario with 50 mobiles nodes. The mobile nodes act as vehicles and can move with a speed up to 20m/s. If two nodes can communicate according to the transmission range, a line is traced by the tool to show that the corresponding communication link is active.

All the aforementioned features that were added to the CANUMobisim framework allow creating more realistic mobility traces easily and rapidly, but can also allow drawing simulation environments with obstacles. To model mobility in presence of obstacles, one could follow the same idea of the Voronoi graph in [2] and create a graph on which nodes will automatically move in a way such as not to pass through obstacles. However, when the graph has too many vertices and edges, this method is very much time consuming and will also result into a very lengthy XML file. To solve this problem, we proposed to use a path finding algorithm. In the next section, we will present the algorithm we developed and the new obstacle mobility model included in AMADEOS.


RealScenario
File Edit View tools CanuSim Window Help
C:\AAkadtest\Montreal downtown.env

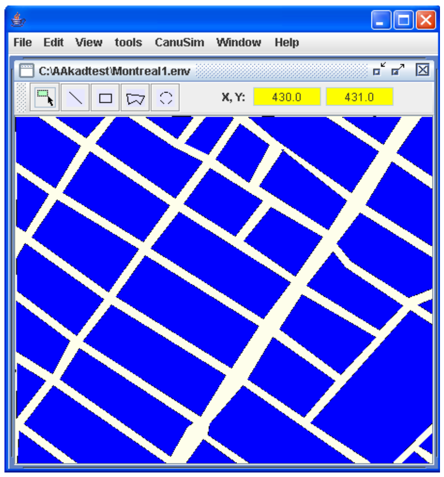

File Edit View tools CanuSim Window Help
C:\AAkadtest\Montreal1.env
X, Y: 430.0 431.0

Figure 3: Image map of a landscape in Montreal Figure 4: The equivalent map with AMADEOS

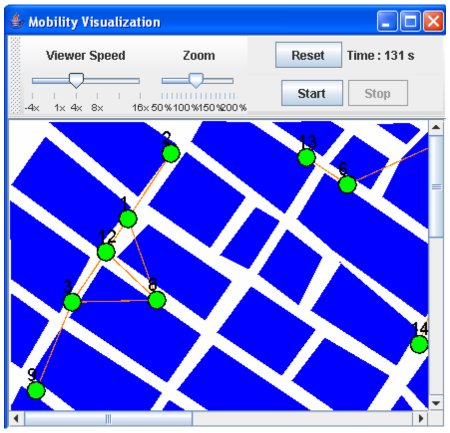


Figure 5: Nodes mobility traces visualization

4. A NEW MOBILITY MODEL FOR ENVIRONMENTS WITH OBSTACLES

The new developed mobility model does not rely on a graph as in [2]. To move from a source location to a destination one, each node has to find its appropriate pathway through the environment which may comprise obstacles. In order to allow nodes not to pass through obstacles, we implemented a path finding algorithm. This algorithm uses a ray launching technique coupled with an optimized sweep line algorithm for the fast ray intersection computation.

4.1 The path finding algorithm

A path is a set of location points which form adjacent segments and no segment intersects with an obstacle in the environment. Our path finding algorithm is described in the diagram of Figure 6. When a node needs to move from its location to a chosen destination location, a ray is launched from its actual position towards the destination position. We repeat this operation each time we want to find the next position to be added to a mobile node pathway towards its destination location. At the beginning (step 0), the start position and the actual position are the same. At step 1, we launch a ray from start to destination and search for the first obstacle hit by this ray. At step 2, we add the first hit point to the path and try to border this first obstacle. To do so, we search for the first edge hit in this obstacle. If an edge is hit, the actual position moves to the intersection point on this edge. At step 3, we choose the nearest side of the hit edge in order to minimize the final path length. We repeat the step 2 until the obstacle is encountered. This means that the ray from the node position to destination does not hit any edge of this obstacle. When the first obstacle is encountered we return to step 1 and search for the next obstacle hit until the destination is reached (i.e. the actual node position and destination are the same). The described algorithm, although simple, can nevertheless still take a lot of computation time because it must compute many intersection points at each step. An efficient optimization method needed to be developed.

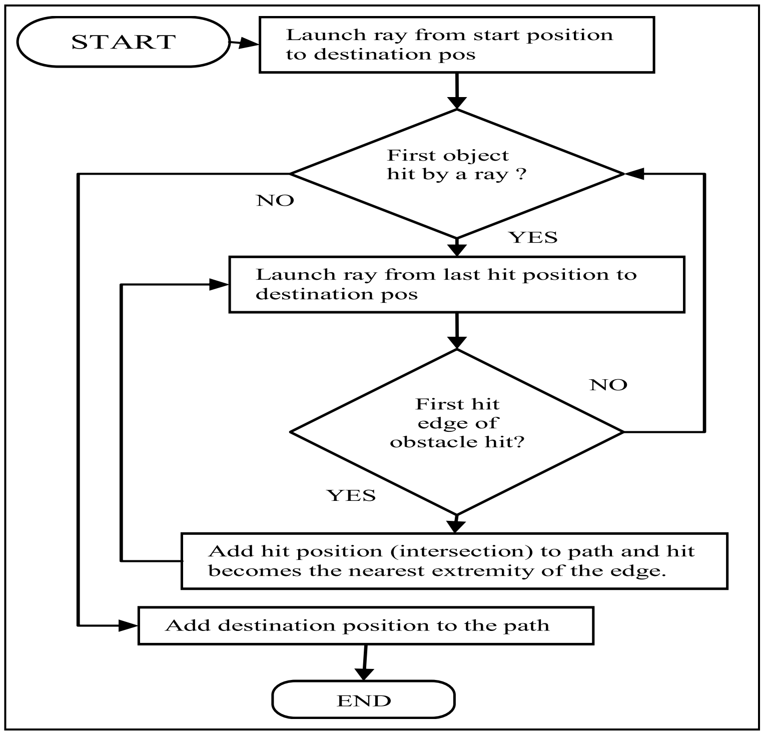

START
Launch ray from start position to destination pos
First object hit by a ray ?
NO
YES
Launch ray from last hit position to destination pos
First hit edge of obstacle hit?
NO
YES
Add hit position (intersection) to path and hit becomes the nearest extremity of the edge.
Add destination position to the path
END

Figure 6: Path finding process flow

4.2 Optimizing intersection search

In order to compute a fast intersection search of a ray with obstacles, we use a sweep line algorithm [7]-[8]. This algorithm has a complexity of $n\log(n+k)$ with k being the number of intersections and n the number of vertices of all obstacles. As for the standard sweep line algorithm [8], our algorithm has two data structures which are maintained during the sweep process. First, a status structure which stores the list of events such as add/remove a line segment from the list or add an intersection is updated at each step. Second, a priority queue which stores the sequence of line segments in the environment is maintained. The line segments are stored in a chosen order. To compare two points we first examine the X coordinate, then the Y coordinate. In our case, as shown in Figure 7, we order segments from left to right by choosing their right extremity point (the segment's extremity whose X coordinate is the highest).

To speed up this algorithm, the simulation environment is clipped to a small rectangular zone each time we search a pathway for a node movement. We choose the sub region between the node position and the destination. The path finding algorithm is then done only in this sub region because the obstacles out of this selected region are far away and will never be hit by a ray. A second performed optimization is to store in a list, the index of the first line segment situated on the left of each point coordinate. Without this information, the algorithm has to restart the sweep process at the first index in the priority queue each time its searches for an intersection. This list of indexes of line segments allows accessing the priority queue randomly which is faster than sequentially.

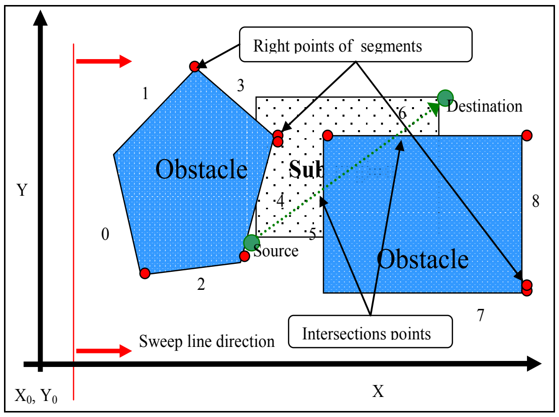


Figure 7: Sweep line intersection

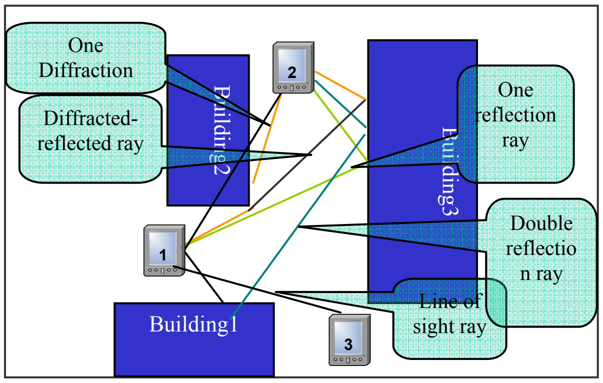


Figure 8: Example of multipath components taken into account for ray tracing

5. RAY TRACING PROPAGATION MODEL

As part of AMADEOS, a module allows modeling radio propagation in areas comprising obstacles. Many accurate ray tracing algorithms exist in literature [16], [18], [19], [21], but these cannot be directly used in MANET simulations because the involved computations are too much time consuming. The path loss at a point has to be updated each time the antenna position changes, which increases the computation time exponentially. Most of existing ray tracing algorithms need many hours to run. In [16], the researchers developed a 3D ray tracing method which approximates experimental measurements, but the approach requires very powerful machines as well as the use of multi processing capabilities. In [18], the authors developed a ray tracing method for NS-2 that needs a pre-processing task which can take more than ten hours to complete. After, the pre-processing task, the algorithm uses statistical markov chains to compute the ray tracing, which means that the method is not purely deterministic. In AMADEOS, the aim was to propose a calculation method that speeds up the ray tracing by using some assumptions that do not affect too much the calculation accuracy.

5.1 AMADEOS ray tracing model

The proposed ray tracing method uses geometrical optics to trace the LOS (Line Of Sight) and the reflected signal components. We choose to compute the reflected rays based on the image method [22]. Diffraction from building corners is accurately characterized using the uniform theory of diffraction [15] .The strength of each component is derived explicitly based on the position and the material dielectric properties of buildings. In general, the LOS and reflected paths provide the dominant components of the received signal since diffraction and scattering losses are high. However, in regions close to scattering or diffracting surfaces, which are typically blocked from the LOS and reflecting rays, these other multipath components may dominate. AMADEOS algorithm computes reflections from building walls and single diffractions from building corners. Multipath components with double diffraction and double reflection are also taken into account. In the other cases, the contribution of the multipath components can be neglected in the resultant electromagnetic field strength. An example of all multipath components taken into account by AMADEOS is shown in Figure 8. The field strength resulting from the received signal is obtained from the superposition of all the multipath components. Thus, if we have a direct ray (LOS), and reflected and/or diffracted rays, the expression of the resulting field at the receiver is:

$$\vec{E}_{recv} = \vec{E}_{Line\_Of\_Sight} + \sum \vec{E}_{reflected} + \sum \vec{E}_{diffracted} \quad (1)$$

where $\vec{E}_{Line\_Of\_Sight}$ is the electromagnetic field of the line of sight ray if it exists; $\vec{E}_{reflected}$ is the electromagnetic field of a reflected ray and ; $\vec{E}_{diffracted}$ is the electromagnetic field of a diffracted ray.
The transmitted and received power expressions of an electromagnetic field which is propagated in free space are respectively:

$$P_t = \frac{{E_0}^2}{\eta} A_e \quad (2)$$

$$P_r = \frac{{E_{recv}}^2}{\eta} A_e \quad (3)$$

where $A_e = \frac{\lambda^2 G}{4\pi}$ is the aperture area of the respective antenna, G is the gain, λ is the wave length, η is the impedance of free space given by: $\eta = \sqrt{\mu_0 / \varepsilon_0}$ (4) where $\mu_0 = 4\pi.10^{-7}$ *and* $\varepsilon_0 = 8.85.10^{-12}$ are respectively the permeability and permittivity of air.

The signal power of a radio waves decays when propagating form the source to the receiver. This attenuation also called the path loss is the ratio of the received power under the transmitted power. If the two nodes have the same antenna gain, the total path loss can be expressed as:

$$pathloss = \frac{P_r}{P_t} = \frac{E_{recv}^2}{E_0^2} \quad (5)$$

where P_r and P_t are respectively the received power and the transmitted power of the signal, E_{recv} and E_0 are the associate electromagnetic fields. During simulation, P_r is compared to a threshold in order to consider if a packet can be received or not at the receiver.

The path loss depends on the dielectric properties of objects in the simulation environment. Buildings walls are made of several materials (concrete, brick, various types of glass, etc.). However, different buildings in a given area are generally made of similar materials. In this first version of AMADEOS, we assume that all building walls have the same value for their electrical characteristics. We considered that the walls are made with dry brick blocks whose electric relative permittivity and conductivity are respectively $\varepsilon_r = 5$ and $\sigma = 10^{-4}$ [S/m].

5.2 Speeding up the ray tracing algorithm

Since all node heights generally considered in ad hoc networks are below the height of urban building and houses, the effect of buildings rooftop reflections or diffractions can be neglected. We can then assume the simulation terrain flat and use a 2D ray tracing technique. Many calculation optimizations performed in AMADEOS are based on this assumption. AMADEOS takes also advantage of the intersection search optimization explained in section 4.2. Also, depending on the distance between two nodes, one can predict that the power of some rays may fall deeper under the threshold value. For example if the distance is more than a half of the transmission range, double diffractions and double reflections can be neglected. Furthermore, if the distance is more than three quarters of the transmission range, one can predict that only the existence a line of sight ray can maintain the received power above the threshold. Finally, during simulation, AMADEOS creates a matrix which stores the last path loss between each couple of nodes which are communicating. For a given node couple, the last path loss value stored can be used if the two nodes are not moving or their new positions approximate the old ones according to a certain distance error. To maintain certain accuracy, this matrix is updated each time the position of a node changes by the distance λ or $\lambda/2$ (λ: wave length). If the simulation scenario has small speeds and big pause times, this method can reduce considerably the simulation computation time.

5.3 Algorithm evaluation test

The time consumption problem is known to be a very crucial factor in ray tracing simulations. Figure 9 below shows the multi path rays computed between **node 1** and **node 2** on Figure 8, and the corresponding computation times. The computation time is measured based on the use of an Intel PIII processor running 850 MHz with 384 Mo RAM. When it is possible to compute all multipath rays, the CPU time is about 60 ms, but generally during a simulation it is not necessary to compute all rays, so the computation time is usually smaller than 60 ms and sometimes null (no delay, if the last path loss in the matrix is still valid). For scenarios with 100 nodes or less, we notice that the CPU time is about seven times the time when using basic propagation models in NS-2. For example, a simulation which takes four minutes to run with the two ray ground model [23], will take about half an hour with AMADEOS algorithm. This value is many times smaller that that taken by all other ray tracing algorithms mentioned earlier, as these will typically take many hours to run for a single simulation and will require powerful computers.

Model	**Multi path components rays**	E_{recv} **Complex (a, b) / Power dB**	**Total** E_{recv} **/ Power dB**	**Computation time**
Two ray ground	Line of sight	(5.79e-05,-5.0e-05)/ -82.32 dB	(5.79e-05,-5.0e-05) -82.32 dB	no delay
Ray Trace	One Diffraction	(2.462e-06,-2.61e-06)/-108.9 dB	(3.067e-05,-1.623e-05) Power: -89.1927 dB Total time: 0.0582 s	0.0108 s
	One reflection 1	(2.606e-05,-3.145e-05)/ -87.79 dB		0.0163 s
	One reflection 2	(2.126e-06,1.757e-05)/ -95.04 dB		0.0175 s
	Diffracted-reflected	(3.449e-08,8.683e-09)/ -148.98 dB		0.0136 s

Figure 9: Values of multipath components between node 1 and node 2.

The tests run with AMADEOS ray tracing method show that the path loss computed between two nodes increases by 6-16 dB per reflection and 14-30 dB per diffraction at the frequency 2.4 GHz. These theoretical values are consistent with the existing path loss prediction in [19]-[20] and [21].

5.4 Integrating the ray tracing model to NS-2

NS-2 [23] core is composed of sets of C++ or OTcl classes that enable an object-oriented approach. AMADEOS implements sub-classes that override the propagation model as shown in Figure 10. The existing models in NS-2 are shown in Figure 10: the two ray ground mode, implemented by the TwoRayGround class, the free space model implemented by the Freespace class and the shadowing model implemented by the Shadowing class. In literature, the most widely used propagation in ad hoc simulations is the two ray ground model. This model combines the line of sight ray (free space model) and the ground reflection ray [14]. AMADEOS ray tracing algorithm was implemented as a new class named RayTracing.

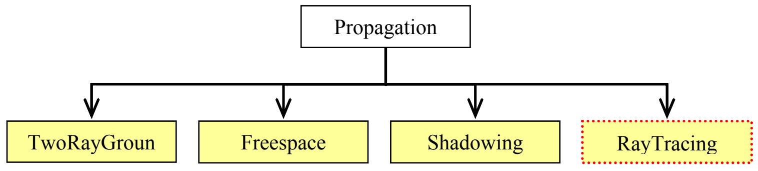


Figure 10**:** Diagram of NS-2 class hierarchy for propagation models

CONCLUSION

Most of MANETs simulation scenario models used in literature in the past few years do not approximate accurately common real world environments. Realistic scenario models are needed in order to assess more accurately ad hoc communication protocols. The work presented here introduces a new tool for facilitating the generation of realistic mobility traces that take into account obstacles. This new tool, called AMADEOS is used as an extension of the CANUMobisim framework. A corresponding ray tracing propagation model was also developed for the proposed mobility model and integrated to NS-2. Using realistic simulation scenarios leads to more accurate results during simulation tests. The use of realistic scenarios can speed up research in the MANETs research area, thus facilitating the standardization of the corresponding protocols and therefore also facilitate a wide commercial deployment of this wireless technology.

REFERENCES

[1] Illya Stepanov, "*A framework for user mobility modeling*" Project page available: http://canu.informatik.uni-stuttgart.de.
[2] Amit Jardosh and Elizabeth M. Belding-Royer, "*Towards Realistic Mobility Models for Mobile Ad hoc Networks*", in Proceedings of MobiCom '02, pp. 217-229.
[3] Yoon J., Liuet M., Noble B. "*Random waypoint considered harmful* ", Infocom 2003, San Francisco
[4] Stepanov I., Tian J., Hähner J., Becker C., and Rothermel K., "*Graph-Based Mobility Model for Mobile Ad Hoc Network Simulation*" , San Diego, California ,April 2002.
[5] Bettstetter C., "*Smooth is Better than Sharp: A random Mobility Model for Simulation of Wireless Networks* " ,Proceedings of the 4th ACM International Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWIM), Rome, Italy ,July 2001.
[6] V. Davies, T. Camp, and J. Boleng "*A Survey of Mobility Models for Ad Hoc Network* ",Special issue on Mobile Ad Hoc Networking, April 2002.
[7] Ivan J. Balaban, "*An optimal algorithm for finding segment intersections*" Proceedings of the 11th Annual ACM Symposium of Computational Geometry. pp. 211-219. 1995.

[8] J. Bentley and T. Ottman. "*Algorithms for reporting and counting geometric intersections*", IEE Transactions on Computing, C-28. pp. 643-647. 1979.
[9] Eric W. Weisstein. Voronoi diagram. http://mathworld.wolfram.com/VoronoiDiagram.html, 2004.
[10] DXF, Autodesk Drawing eXchange Format, http://www.autodesk.com/
[11] *Geographic Data Files* (GDF) Available at (last visited on Mars 1st 2005) http://www.ertico.com/links/gdf/gdf.htm .
[12] Nexus Project, available at http://www.nexus.uni-stuttgart.de (last visited on Mars 15th 2005).
[13] Scan2CAD, Converts scanned drawings into DXF files http://www.softcover.com/ .
[14] Andrea Goldsmith, "*WIRELESS COMMUNICATIONS course*", Stanford University, 2004.
[15] McNamara DA, Pistorius CWI Malherbe JAG, "*Introduction to the Uniform Geometrical Theory of Diffraction*". Artech House, Norwood, MA.1990.
[16] Bohacek S., Sridhara V. and Kim, J.Y. "*The UDel Models - MANET Mobility and Path Loss in an Urban/Suburban Environment*", University of Delaware's ECE/CIS, 2002.
[17] Michael Welzl , NS stuff , http://www.welzl.at/research/tools/ns/index.html
[18] Dricot, J. and De Doncker P. *"High-accuracy physical layer model for wireless network simulations in NS-2"*, IWWAN 2004.
[19] H.M. El-Sallabi, "*Fast path loss prediction by using virtual source technique for urban microcell*", IEEE VTC 2000-Spring Tokyo, pp. 2183 – 2187.
[20] Schenk T.C.W., Bultitude R.J.C., Augustin L.M., Van Poppel R.H. and Brussaard G., "*Analysis of propagation loss in urban microcells at 1.9GHz and 5.8GHz*".
[21] Erceg V., Rustako A.J. and Roman R.S., "*Diffraction around corners and its effects on the microcell coverage area in urban and suburban environments at 900MHz, 2GHz, and 6GHz*", IEEE Transactions on Vehicular Technology, vol. 43, no. 3 pp. 762 – 766, Aug 1994.
[22] Henry L. Bertoni, "*Radio Propagation for Modern Wireless Systems*". Prentice Hall PTR, 2000.
[23] The network simulator, NS version 2, http://www.isi.edu/nsnam/ns/ .
[24] GloMoSim: Global mobile information systems simulation library, http://pcl.cs.ucla.edu/projects/glomosim/.
[25] AMADEOS: http://www.gel.usherb.ca/interlab/downloads/amadeos.html.