3D Ray-Tracing Parallel Model for Radio-Propagation Prediction

André Mendes Cavalcante, Marco José de Sousa, João Crisóstomo Weyl Albuquerque Costa,
Carlos Renato Lisboa Francês, Gervásio Protásio dos Santos Cavalcante and Claudomiro de Souza Sales Jr

***Abstract*— A computational parallel model based on 3D ray-tracing for radio-propagation prediction is presented. This approach considers that the main tasks in a 3D ray-tracing technique can be evaluated in an independent and/or parallel way. The workload distribution among the participant nodes of the parallel architecture (cluster of PC's), is performed through a random assignment of the initial rays and the field points for them. Simulations are realized in order to validate and evaluate the performance of the proposed model.**

***Index Terms*— Parallel computing, cluster of PC's, 3D ray-tracing.**

I. Introduction

The great growth in mobile communications needs fast and accurate prediction of radio wave propagation for system deployment. Such predictions can represent an important role in determining network parameters including coverage, transmitted-data rates, optimal base station locations, and antenna patterns. In this context, ray-tracing based radio propagation prediction models have shown promise, mainly in modern radio wave propagation environments [1]-[5]. Although ray-tracing approaches are very useful in the design, analysis, and deployment of wireless networks, it has been recognized that these models are computationally very expensive and require a considerable amount of processing time to attain reasonable accurate prediction results [2],[4].

Several approaches have been proposed to shorten the computation time for ray-tracing prediction models. In [2], the complexity of the building databases was reduced by simplifying footprints. Data filtering and cleansing techniques have been proposed in [5]. In order to address the same problem, some procedure approximation methods are also employed in [6]. All these approaches have a common trade-off: they trade prediction accuracy for processing time. A natural way to overcome the above trade-off is to use the parallel and/or distributed computing techniques to speed up computations, while keeping the accuracy intact [4]. More specifically, the usage of a cluster of PC's (sometimes referred as a class of COW's - Cluster of Workstations) is particularly attractive as such computer system configurations are readily available at this time.

Recently, some parallel computational strategies have been proposed in order to reduce the required computational time without affecting the prediction accuracy requirements [4],[7].

The authors are with the Applied Electromagnetism Laboratory (LEA), Federal University of Pará (UFPA), Zip Code: 60075-900, Belém-PA, Brazil. email: amc@ufpa.br

In [4], the strategy of parallelization proposed is very complex and hard to implement. This approach was applied in a 2D ray-tracing model, being also tested in a kind of 3D ray-tracing that have some restrictions in the diffraction mechanism (Vertical Plane model - [2]). Both model versions are very dependent on the SBR algorithm implementation. In [7], the parallel computational strategy was applied only to the ray-processing stage. The parallel model proposed in this paper is schematized to the overall process, being very simple to implement computationally and can be applied easily in full 3D ray-tracing channel model without any diffraction restrictions (if desired). This new approach allows to reduce or even eliminate many restrictions early imposed in ray-tracing models by practice reasons (high computational cost), favoring to improve the accuracy and a possibility of incorporating new propagation mechanisms, such as diffuse scattering [4],[8] and propagation in forest environment [9]. Additionally, it allows analyzing more complex structures (scenes).

This paper is organized as follows. Section II discusses the baseline ray-tracing based radio propagation prediction algorithms and presents a brief description of the full 3D SBR model adopted in this paper. Section III outlines the parallel computational model proposed. Section IV presents some simulations in order to validate the parallel ray-tracing model. Conclusions are made in Section V.

II. Ray-Tracing Techniques

There are basically two approaches for tracing of rays in the radio-propagation context: the first one is based on Image Theory (IT) [10]. Such approach is strongly dependent on size and complexity of the environment; it has been more used in small and simple environments involving only reflections [10]. Some authors have related the possibility of incorporating diffraction and transmission points searching algorithms in such approach, but even so, its intrinsic limitation early mentioned remains. On the other hand, the shoot-and-bouncing-ray (SBR) method (referred sometimes as "Brute-Force") is the ray-tracing approach more suitable for large and complex environments [10], involving any combination of basic interactions (reflection, transmission and diffraction). The intrinsic limitation of this approach is the high required processing time in order to evaluate all raypaths.

Independent of the adopted approach, the spent computational time for the program execution can reaching very large values [4], mainly to environments inserted in the modern configurations of wireless communication system (*e.g.*, with multiple sources, outdoor-indoor interactions, etc).

The first natural effort in order to reduce the processing time in such models is optimizing the ray-face intersection tests (or shadowing tests). There are several approaches related to that optimization, such as BSP (Binary Space Partition), SVP (Space Volumetric Partition), Angular Z-buffer algorithm, BV (Bounding volumes) and so on [10]. Additionally, efforts have been made to parallelize the code that implements the ray-tracing algorithm [4],[7]. In IT, the parallelization of the code is not trivial, since the data structure used in this technique (tree of images) is totally concatenated, hindering the splitting of the tasks and the load balancing among the processor nodes. On the other hand, the SBR technique is already intrinsically parallel, because the rays that are launched by the transmitting antenna are independent from each other, allowing the SBR code to be directly applicable in the parallel programming paradigm. Therefore, the parallel model was developed over a SBR approach.

A. Full 3D SBR Technique

According to the classical SBR technique, rays are launched by the transmitting antenna in all directions, each one presenting a wavefront portion that propagates from the antenna. SBR methods are also referred as forward methods, due to the simulation to be performed from the transmitting antenna, tracking the path of each ray and their descendants (new generated rays by interaction with faces of the scene). In this paper, such technique was implemented into three stages labeled as: ray-launching, ray-reception and ray-tracking.

- **Ray-launching stage:** This stage is responsible for the launching of rays from the source (transmitting antenna). The efficiency of this stage is measured by the ability in generating a uniform launching of rays by the source in the space in order to subdivide wavefronts with nearly equal shape and area. In 2D methods, this requirement is perfectly obtained. However, in 3D methods, an equal division of the wavefront originating from the transmitting antenna among rays is not trivial and it requires special procedures. An efficient 3D source modeling strategy that has been used largely in literature and was adopted in this paper is the one presented in [11]. In this approach, the ray-launching at the transmitter (source) is based on a technique where a regular icosahedron is inscribed in a unit sphere surrounding the transmitting antenna. Each icosahedron vertex represents the launching direction of a ray. In order to provide more rays, each triangular face of the icosahedron is further divided into smaller triangles, according to illustrated in Fig.1. However, this subdivision into more facets is an approximate method in the sense that the angular separation α between neighboring rays is no longer exactly the same. This is because the new generated vertexes are not all on the circumscribing sphere of the original icosahedron [12]. These results present in about a 20% difference between the maximum and minimum value of α [12]. Thus, the ray-launching model adopted here will bring a value to the angular separation α changing between (α_{min}, α_{max}). In the ray-reception stage (described later), it is necessary to define a fixed angular separation in order to discover which components (rays) are received in each field point. If it is adopted to the angular separation, the value defined by α_{min}, some multpath components that should be taken into account will not be. Otherwise, if α_{max} is adopted, there will be multiple counting of components. The ray-launching strategy implemented in this paper always considers the value of α_{max} in all calculations, being the multiple counting problems solved by a simple additional procedure briefly described in the next stage.

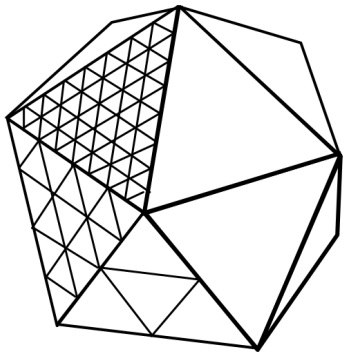

Fig. 1. Recursive subdivision of the faces of a regular icosahedron.

- **Ray-reception stage:** This stage determines if a certain ray must be considered as a received one to the reception points (field points). The strategy of reception adopted here is based on an adaptive reception sphere, according to the model described in [11]. This strategy works well if the angular separation α between neighboring rays is exactly the same. As mentioned early, this is not achieved in a tri-dimensional implementation. To the angular separation case considered in this paper (*i.e.*, α_{max}), this reception strategy undergoes some problems related to multiple counting of components. Such problems are solved by a simple procedure that verifies if a raypath with the same mechanisms on the same objects already was computed in simulation. In the affirmative case, only the closest raypath of the reception point is stored. The verification procedure is performed through a comparison of the *strings* that define the raypaths, being then, a computationally efficient procedure, not generating considerable workload in this stage.
- **Ray-tracking stage:** This stage is responsible to track the paths of each ray based on its interaction with scene obstacles. For each ray launched by the source, a recursive algorithm is performed in order to verify if the ray intersects some scene obstacle (face or edge) or some field point (reception sphere). If the ray intersects a face then the source ray is replaced by a reflected ray and by a transmitted one (for outdoor scenes, the transmitted ray is neglected by our SBR model), to which the recursive algorithm is applied again. If the ray intersects some field point, a ray-reception procedure (described in the previous item) is trigged. In the last case, if the ray intersects an edge (details described below), the ray source is replaced for several diffracted rays.

The number of generated diffracted rays depends on the desired resolution. In our model, it is considered that diffracted rays are generated with an angular separation fixed in ($2\alpha_{max}$) for the first diffraction, ($3\alpha_{max}$) for the second and so on. This assumption (resolution decrease of diffracted rays according to its order) reduces dramatically the memory and processor usage and not affects substantially the prediction accuracy if it is adopted a high initial resolution to α_{max}.

A problem faced by full three-dimensional techniques in the ray-edge intersection procedure is the identification of "illuminated" edges and calculation of the respective diffraction point. In order to perform this procedure in 3D space, it was created an edge reception cylinder concept. This cylinder is built with its longitudinal axis exactly over the edge and presents two semi-spheres, one in the top and another one in the base (it is equivalent to serial reception spheres overlapped along the edge), according to schematized in Fig.2. The adaptive radius *R* of the reception cylinder was considered as the same given by the adopted reception sphere model. If a ray intercepts the reception cylinder of some edge, this one will be identified as a diffracting potential edge. After this identification, it is performed a procedure that checks if this edge will really be an effective diffracting edge. In the affirmative case, the approximated diffraction point is determined. Such procedure depends on the history of the ray that intercepts the cylinder (*i.e.*, the mechanism how it was originated) and is solved making use of a combination of Image Theory (IT) [10], diffraction law (Keller's cone) [10] and generalized Fermat's principle [12].

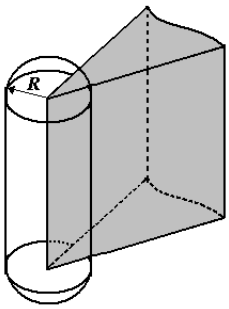

Fig. 2. Concept of edge reception cylinder.

III. Parallel Model

The proposed parallel model for SBR algorithms was schematized in three stages, according to shown in Fig.3: *Pre-Processing*, *Processing of Rays* and *Post-Processing* ones. The basic idea of this model is that after a data pre-processing phase, the total workload can be divided among the nodes that compose parallel architecture (cluster of PC's), through a random distribution among them, of the initial rays to be launched and field points (reception points) to be evaluated. The efficiency of this approach is guaranteed by the independence of the involved entities (rays and field points) and by the form of the employed distribution (random). The random approach tends to be more efficient in the load-balancing issue as larger the total number of emitted rays and the field points are, exactly the case that most justifies the use of parallel computing [7]. Through this strategy, the processing load of a homogeneous cluster is balanced through the distribution of the equal number of initial rays and field points (randomly chosen) for each node. For a heterogeneous cluster, the rays and field points number of each node must be proportional to its processing capacity. Evidently, discovering the processing capacity of computers may be done previously, being it even possible to estimate it based on characteristics of hardware and software.

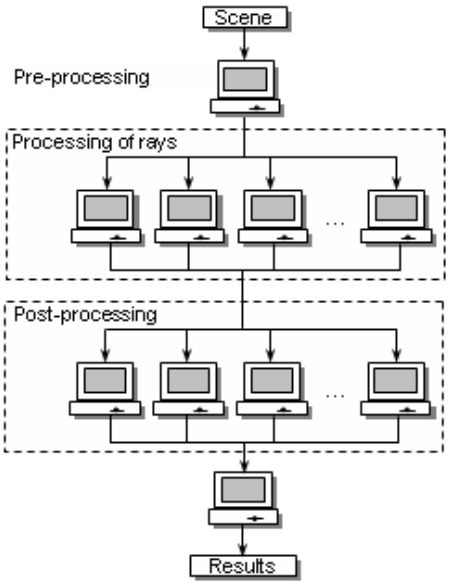

Fig. 3. Model of parallelization for the 3D SBR algorithm.

In the context of the parallel and distributed computing, the proposed model can be inserted in the SPMD (Single Program Multiple Data) paradigm [13], because for a specific scene, each node performs the same SBR program over distinct data (initial rays and field points). The initial communication strategy among the nodes in order to supply such input data through the network could be implemented, for instance, using MPI (Message Passing Interface) standard communication library [14]. However, a simplest strategy was implemented, where customized input files for each node are previously created and distributed through a network file system, as the NFS (Network File System), used in UNIX systems [15]. After the generation and loading of the input files (*Pre-Processing* stage), each node will perform the processing of the rays defined for it. When the rays simulation is over, each isolated process (node) can send their results through the network using MPI, or make available in the form of local file shared through the NFS (in this paper, the NFS strategy was adopted). The processing of rays result is a report of all the rays that reached the field points defined in the scene. The reception and organization of results provided by each node consist in the post-processing stage.

To proceed, each stage will be briefly detailed, being adopted the terminology that if the program is performed in a serial way, it is called as *serial mode* and if the program is performed in a cluster, it is called *cluster mode*:

1) **Pre-Processing stage:** This stage consists basically on the definition and creation of customized input files for each node. After this step, two files for each node, designed as *setup file* and *rays file*, are defined. Such files and their usage will be explained in subsequent procedures listed below:
   a) ***Loading of the setup file*:** This procedure consists on reading of a predefined *setup file*. A *setup file* contains information about simulation parameters (transmitted power, antennas type, maximum number of interactions, field points locations, etc) and the names of the scene (building database) and rays files that should be loaded.
   b) ***Loading of the rays file*:** It consists on reading of the *rays file* defined in the *setup file*. A *rays file* contains the directors of the initial rays that were randomly assigned to a certain node. In the *serial mode*, all directors are assigned to a single node.
   c) ***Generation of table files*:** This procedure consists on generation of table files for each node. A *table file* contains a list of field points that will be under responsibility of a specific node. This assigning of field points is performed by a random way and so that each file contains a approximately equal number of distinct field points. Theses files will be useful in the parallelization of the *Post-Processing* stage (additional details will be given later). As implementation proposal in the *cluster mode*, it is chosen a specific node to be responsible by the generation of a *table file* for each node (including itself). In serial mode, all field points are assigned for a single node.
2) **Processing of Rays stage:** This stage is responsible to perform the 3D SBR algorithm described in the Section II-A. Each node is responsible to perform this procedure only for the rays defined in its *rays file*.
3) **Post-Processing stage:** It consists basically on the saving of path files and results evaluation. Three steps are defined in such stage, according to what is listed below:
   a) ***Saving of path files*:** In this step a file (*path file*) for each field point is created containing information about all raypaths that reached it. In the cluster mode, each participating node generates their self files for each field point, requiring an additional procedure for assembling these files in order to mount a single file per field point.
   b) ***Synchronization Saving/Assembling*:** In order to begin the *Assembling of files* step, it is necessary that all the nodes have already concluded the saving procedure (*Saving of path files* one). To each node indicates for the other nodes that its saving step is over, a *status file* is created. Such file does not contain any information and is used only to status check (for instance, if the *status file* of a specific node was created, then it indicates that the node is ready to begin the assembling procedure, otherwise, the node is still not ready). When all nodes were ready (synchronization phase), they begin to perform the *Assembling of files* step.
   c) ***Assembling of files***: In this step, each node is responsible for assembling the path files of the nodes just for the field points defined in its *table file*. It is important to note that in this step is necessary only in the cluster mode.
   d) ***Synchronization Assembling/Evaluation*:** In order to perform the subsequent step (*Results Evaluation* one), it is necessary that all the nodes have already concluded the assembling of files procedure. In a same way to realized in the previous synchronization step, each node creates a *status file* to indicates for the other nodes that its *Assembling of files* step is over. When all nodes were ready, they begin to perform the *Results Evaluation* step.
   e) ***Results Evaluation*:** This step is responsible for the prediction results evaluation (electric field, received power, arrival angle for each ray, etc) and generation of the output files with such information for each field points. In the cluster mode, each node is responsible for evaluating results just for the field points defined in its *table file*.

IV. RESULTS

In order to validate the proposed parallel model, it was considered as study of case an outdoor scene in Ottawa city (Canada). The scene considered is within the 1000 m x 600 m area according to shown in Fig.4. The transmitter was located in the position labeled as “Tx” at a height of 8.5 m and the field points were placed along the Laurier street at a height of 3.65 m (Fig.4). All antennas were vertically polarized. As ray-launching algorithm it was considered the full 3D SBR model briefly described in Section II-A together with the UTD (Uniform Theory of Diffraction) presented in [10]. The fields were calculated at a frequency of 910 MHz considering raypaths up to 4 reflections and 2 diffractions. The effects of paths that diffract over the rooftops were neglected due to the transmitting and receiving antennas were located right below the building heights (according to information reported in [16]), and in these situations, such paths are usually of negligible power compared to other paths that propagate among the buildings [9]. Although this supposition, the proposed full SBR 3D model was still performed. The building data for the calculations were obtained directly from the maps in [17] which contained the footprints of the buildings. In [16] no information about the terrain was reported, being assumed a flat terrain in all calculations. Following the suggestion in [17], it was set the relative permittivity of all the building walls to 6, and the conductivity to 0.5 S/m. A relative permittivity of 15 and a conductivity of 0.05 S/m were used for the ground.

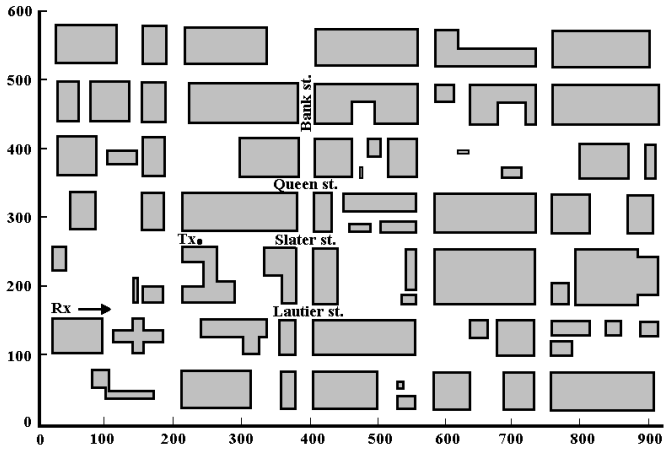

Fig. 4. Map of Ottawa showing street names, transmitter and receivers locations

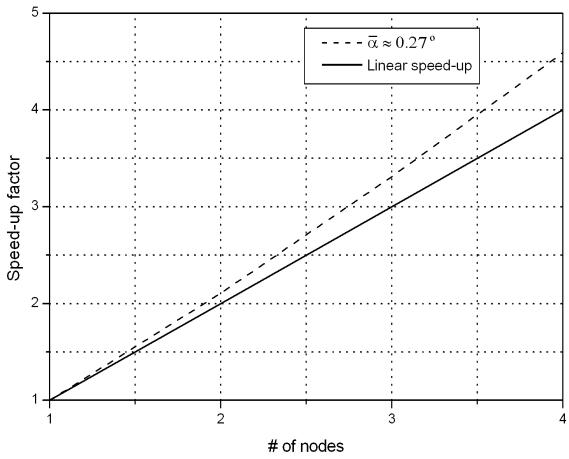

Fig. 5. Speed-up factor to 655362 rays launched by the source

The simulations were carried out under a cluster consisting of four (04) nodes with Pentium IV HT 3.2 GHz processors and main memory of 1.0 GB. All computational code was implemented using C++ object oriented software language. The used compiler was the g++ version 3.3.5 20050117 (pre-release) under a GNU/Linux operating system. The master/slave paradigm was used in order to implement the Unix network file system (NFS - Network File System). Customized input files (setup file and rays file) to each process (node) were previously constructed. All those files jointly with the scene file were distributed on the network through the Unix NFS. To evaluate the performance of the proposed parallel model, some interesting metrics were adopted, as speedup, workload expansion ratio, and resource utilization [4]. Considering that T_{seq} is the best finish time achieved when only one machine is used (serial mode), t_i is the finish time for the ith node when a *n*-node cluster configuration is used, T_{max} and T_{avg} are the maximum and average finish times, among the *n* nodes, while T_{sum} is the summation of finish times for all nodes, then, $T_{max} = \max_{i=1}^{n} t_i$, $T_{sum} = \sum_{i=1}^{n} t_i$, and $T_{avg} = T_{sum}/n$. The speedup S_n, the workload expansion ratio W_n, and the resource utilization U_n can be computed as $S_n = T_{seq}/T_{max}$, $W_n = T_{sum}/T_{seq}$, and $U_n = T_{avg}/T_{max}$. In order to more accurately measure the scalability of the proposed model, it was employed the metric of efficiency $E_n = S_n/n$ [4].

According to Fig.5, the speed-up factors obtained for the case with 655362 rays launched by the source (*i.e.*, a mean angular separation between neighboring rays in 3D space $\overline{\alpha} \approx 0.27^o$) presented behavior above linear case. This situation is referred in literature as "super-linear speed-up".

Table I shows the number of processed rays (ray-load balancing) by each node in several cluster configurations for 655362 rays launched by the source, presenting a maximum processed rays difference related to average value (number of processed rays in serial mode/ number of nodes) around 1.12%. Additionally, in Table II are showed the mean processing times wasted by each node. Although each node processing a different number of rays, the largest obtained processing time to each cluster configuration was always below to expected ideal average time (serial execution time / number of nodes).

TABLE I
PROCESSED RAYS

Node	Serial	2-nodes Cluster	3-nodes Cluster	4-nodes Cluster
0	124310173	62325589	41107549	30881464
1	-	61984584	41899821	31214554
2	-	-	41302803	31154001
3	-	-	-	31060154

TABLE II
PROCESSING TIME

Node	Serial	2-nodes Cluster	3-nodes Cluster	4-nodes Cluster
0	11226.838 s	5304.846 s	3380.315 s	2439.285 s
1	-	5329.656 s	3390.986 s	2431.279 s
2	-	-	3379.247 s	2445.582 s
3	-	-	-	2432.800 s

Table III shows the performance evaluation metrics applied in the proposed parallel model. According to shown it, the workload expansion ratio obtained in all cases was always below ideal case ($W_n = 1.0$), decreasing its value as the number of nodes increases. It features that it is possible expecting a good scalability of the model. The resource utilization rates obtained are very close to the ideal utilization rate indicating that all nodes spend little time in idle status. The efficiency of the proposed model improved with the increase of the number of nodes, presenting values above ideal efficiency in all considered cluster configurations.

TABLE III
PERFORMANCE EVALUATION METRICS

n-node Cluster	S_n	W_n	U_n	E_n [%]
1	1.000	1.000	1.000	100.0
2	2.106	0.947	0.998	105.3
3	3.310	0.904	0.998	110.3
4	4.591	0.868	0.996	114.7

As the unique difference among the processes of each node is the data volume of the input files, these performance evaluation results show that the processing time of the tasks performed by each node presents a reduction rate above linear related to reduction rate of the handled data volume, mainly in procedures related to the scanning of the used data structures and memory allocating. It implies that if the SBR algorithm is partitioned (i.e., distribution of initial rays and field points among several input files) and it is structured to be performed of serial way, nevertheless it will be more attractive than being performed in serial way with no partition. The scalability of the model is naturally guaranteed due to independence of the initial rays and field points. However, this super-efficiency presented by the model only will be maintained while the speedup gain obtained in the ray-processing stage of each node in a certain cluster configuration is large enough and overcoming the speedup losses generated in the other procedures (mainly in the Assembling files one). This requirement can be achieved increasing the complexity of the scene and/or increasing the resolution of the initial rays to be launched. Besides improving the efficiency model, the increase of these entities (scene complexity and ray resolution) makes the SBR algorithm more accurate.

In order to give some indication of the prediction quality provided by the 3D SBR model along Laurier street, the predicted propagation path loss was compared to measurements reported in [17]. The results are shown in Fig.6.

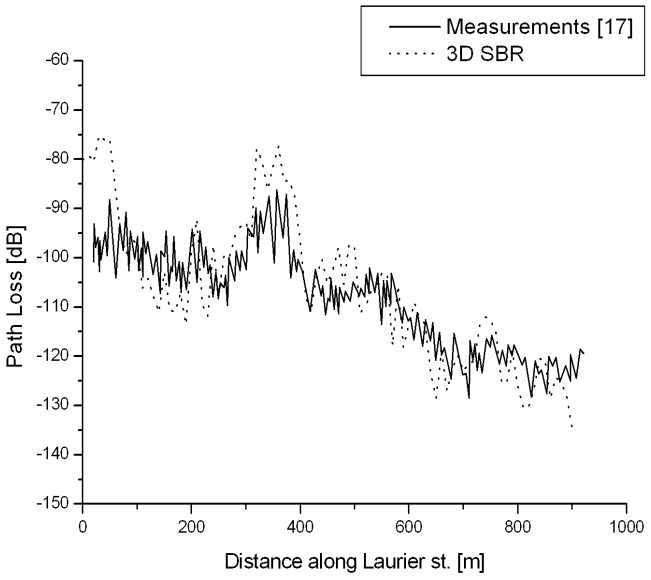

Fig. 6. Path Loss along Laurier st.

The agreement is good considering the quality of the building data and the lack of information on building materials. It is worthwhile to comment on one of the more notable difference with the measurements. The error at the beginning of Laurier St. is surprising because there is nearly a line-of-sight path from the transmitter. An explanation is that probably there are some trees or other obstructions that scatter the signal in such area.

V. Conclusions

In this paper, it was presented a computational parallel model applied in full 3D ray-tracing techniques for radio-propagation prediction. Such approach is based on independence of the tasks in the SBR ray-tracing algorithm in order to efficiently distribute the total workload (by a random distribution of the initial rays and the field points) among the nodes of the parallel architecture (cluster of PC's). Several issues related to practical implementation of the parallel model were described. Some simulation results have been presented in order to demonstrate the efficiency of the proposed parallel model.

Acknowledgments

This paper was partially supported by CNPq and Ericsson project (UFA01).

References

[1] Richter, J., Al-Nuaimi, M.O.; Ivrissimtzis, L.P., "Optimization of radio coverage in urban microcells using a UTD based ray-tracing model," *2004 Antennas and Propagation, IEE Proceedings*, vol. 151, Issue: 3, 21 June 2004, pp. 187 - 192.

[2] Z. Chen, A. Delis, H.L. Bertoni, "Building footprint simplification techniques and their effects on radio propagation predictions," *Comput. J.*, 47 (1) (January 2004) 103-133.

[3] Kipp, R.A.; Miller, M.C., "Shooting-and-bouncing ray method for 3D indoor wireless propagation in WLAN applications," *Antennas and Propagation Society Symposium, 2004*, vol. 2, 20-25 June 2004, pp 1639 - 1642.

[4] Z. Chen, A. Delis, H.L. Bertoni, "Radio-wave propagation predictions using ray-tracing techniques on a network of workstations (NOW)," *J. Parallel Distrib. Comput.*, 64 (2004) 1127-1156.

[5] T. Kurner, A. Méier, "Prediction of outdoor and outdoor-to-indoor coverage in urban areas at 1.8 GHz," *IEEE J. Selected Areas Comm.*, 20 (3) (April 2002) 496-506.

[6] Z. Chen, H.L. Bertoni, A.Delis,"Progressive and aproximate techniques in ray-tracing based radio wave propagation prediction models," *IEEE Trans. Antennas Propagation*, 52 (1) (2004) 240-251.

[7] Cavalcante, A. M., Sousa, M. J., Sales Jr, C. S., Costa, J. C. W. A., Francês, C. R. L., and Cavalcante, G. P. S.,"Computational parallelization strategy applied in full 3D ray-tracing wireless channel modeling," *International Microwave and Optoelectronic Conference - IMOC'2005*, Brasília - DF, Brazil, July 2005.

[8] Degli-Esposti, V,; Guiducci, D.; de'Marsi, A.; Azzi, P.; Fuschini, F.;"An advanced field prediction model including diffuse scattering," *IEEE Transactions on Antennas and Propagation*, vol. 52, Issue: 7, July 2004, pp. 1717 - 1728.

[9] Bertoni, H. L., *Radio propagation for modern wireless systems*, Prentice-Hall - Wireless Communications Series, 2000.

[10] M. F. Cátedra and J. P. Arriaga, *Cell Planning for Wireless Communications*, Artech House - Mobile Communications Series, 1999.

[11] Scott Y. Seidel and Theodore S. Rappaport, "Site-Specific Propagation Prediction for Wireless In-Building Personal Communication System Design," *IEEE Trans. on Veh. Technol.*, vol. 43, no. 4, Nov. 1994.

[12] S.Y. Tan and H.S. Tan, "Modelling and measurements of channel impulse response for an indoor wireless communication system," *IEE Proc.-Microw. Antennas Propag.*, vol. 142, no. 5, October 1995.

[13] Y. Foster, K. Kennedy, J. Dongarra, G. Fox, *Sourcebook of Parallel Computing*, Morgan Kaufmman Pub, 2002.

[14] W. Gropp, E. Lusk, A. Skjellum, R. Thakur, *Using MPI : Portable Parallel Programming with the Message Passing Interface (Scientific and Engineering Computation)*, 2nd Edition, MIT Press, 1999.

[15] Tanenbaum, A.S., *Computer Networks*, 4th Ed.,Prentice Hall, 2002.

[16] J. H. Whitteker, "Measurements of path loss at 910 MHz for proposed microcell urban mobile systems," *IEEE Trans. Veh. Technol.*, vol. 37, no. 3, pp. 125-129, Aug. 1988.

[17] S.Y. Tan and H.S. Tan, "Propagation model for microcellular communications applied to path loss measurements in Ottawa city streets," *IEEE Trans. Veh. Technol.*, vol. 44, no. 2, pp. 313-317, Aug. 1995.