Preconditioned Astigmatic Beam Tracing for Urban Propagation

Emidio Di Giampaolo, Fernando Bardati, and Marco Sabbadini

***Abstract*—In wireless communication network planning, deterministic and statistical propagation tools have been developed. Astigmatic Beam Tracing, a deterministic method based on ray tracing (GO/UTD), can be applied to urban propagation modeling. A numerical code has been applied to a model of urban scenario with increasing complexity. The results of a numerical analysis are discussed to evaluate the relevance of the test case size on the computational charge.**

***Index Terms*—Ray-tracing, urban propagation.**

I. Introduction

WIRELESS communications demand accurate methods for electromagnetic field computation in complex environments such as urban/indoor scenarios. Different propagation models are in use, however ray-optics GO/UTD algorithms are suitable for microcellular environments and small distances from low-height base stations [1].

Based on ray tracing, ray-optics models calculate the total field impinging on a receiver as ray field superposition. In particular, three-dimensional ray-tracers require high-resolution site-specific information, which hugely increases the computational cost. The overall accuracy depends on the amount of details in the scene that a ray tracer is able to handle, as well as on the algorithms in use for ray tracing, several of which have been proposed. Those based on the visibility theory are well suited for electromagnetic propagation studies as they can lose less contributions and can be faster [2]–[5] than other methods such as the image method. Astigmatic Beam Tracing (ABT) [5], [6] computes the visibility in a global manner, in the sense that any area-to-area or source-to-area visibility test is performed in the presence of all the obstacles featuring an environment. This property makes ABT well suited to treat multiple interactions, including diffraction, which can be responsible for communication to shadowed (nonline-of-sight) areas. Filling a data base with the characteristics of sources and obstacles, and performing the visibility computations constitute the off-line steps of ABT. They result in a hierarchical arrangement of ray tubes. ABT adopts a beam-tree for data storage. Ray path determination from a source to an observation point, and field propagation using GO/UTD along ray paths constitute ABT on-line steps. We here report recent improvements in the off-line steps of a computer code, which applies ABT to moderately large urban scenarios, and present numerical results with a discussion of the computational cost.

Manuscript received February 4, 2003. The review of this letter was arranged by Guest Editor Roberto Sorrentino.

E. Di Giampaolo is with the Dipartimento di Ingegneria Elettrica, Università dell'Aquila, 67040 L'Aquila, Italy (e-mail: emidio@ing.univaq.it).

F. Bardati is with the DISP, Università di Roma Tor Vergata, 00133 Roma, Italy (e-mail: bardati@disp.uniroma2.it).

M. Sabbadini is with the TOS-EEA, European Space Agency, 2201 AZ, Noordwijk, The Netherlands (e-mail: marco@xe.estec.esa.nl).

Digital Object Identifier 10.1109/LMWC.2003.815697

II. Scene Modeling and Beam-Scene Intersection

Application of ABT, as any deterministic propagation tool, to urban scenarios requires geometrical and morphological models of the propagation environment as input data. Geometrical data can be obtained from Geographical Information Systems (GIS), which are usually used as database manager and graphical interface. Database information is exchanged between GIS and ABT by a translator, which supports standard graphical formats for scene description. Vector formats are convenient for geometrical data within a Boundary Representation (known as b-rep [7]). We restrict b-rep to handle only polyhedrons whose surfaces are 2-manifolds with convex polygonal boundaries, so that the surface of hills and buildings with curved walls are approximated with polygons. An urban scene is modeled as a unique continuous surface resulting from the union of adjacent flat facets. Each facet is characterized by edges, slope and dielectric properties, while small-scale surface roughness is neglected. The number of facets depends on desired resolution for a given scene.

A beam is a bundle of rays with a finite cross section i.e., a portion of a normal ray congruence. ABT exploits convex beams, which are defined by convex cross-section. When a beam is projected onto a scene, the facet(s) included in the intersection of the beam with the polyhedron representing the scene is determined resorting to a search algorithm based on space partitioning. Then the beam is clipped to the scene facet(s). Therefore, the most performed operation in ABT is polygon-beam intersection. This visibility test is a heavy computational task and dominates the overall cost for large scenes.

Preconditioning is particularly valuable in beam tracing because allows quick and accurate ray beam clipping. To achieve this, each facet stores data about its topology, i.e., adjacent facets to a given facet, in order to allow fast recognition of neighboring facets. A particular data structure, known as winged-edge (WE) [8], has been used in the work reported by this paper to decrease the cost of the queries corresponding to adjacency relationships required to compute beam clipping. The winged-edge data structure is particularly suitable for our problem as it allows determining in constant time which vertices or faces are associated with a specific edge, i.e., the information required to clip beams and to determine the portions of edges involved in diffraction. A further useful property of winged-edge is that

the data structures for edges, faces, and vertices are each of a small, constant size [9], [10]. The winged-edge data structure uses edges as reference elements and to store most of the adjacency information. The data structure associated to each edge contains pointers to the two end-points of the edge, to the two faces adjacent to it and to four of the other edges sharing the two vertices. These four edges are the "wings" from which the structure gets its name. For any given edge on the contour of a face, we are allowed to promptly find all the other edges by following the loop of edge pointers. The coordinates of vertices and the identity of faces for further processing can then be derived form the edge data structures. The winged edge data structure requires a fixed amount of memory to store the data. In Woo [10], it is shown that the total storage cost is $9E$ while the total time complexity is $6EV_i + 3k$. E is the number of edges constituting the scene; EV_i is the number of edges connected to the particular vertex V_i, and has been chosen in [10] as a unit for complexity; k is a constant time for direct access to data. Indeed, of the nine topological queries, three are constant time queries while the other six queries require local "clocking" around a face or a vertex costing EV_i in time. Therefore, EV_i is variable with $3 \leq EV_i \leq E/2$[9], [10].

Besides clipping, a beam tracer algorithm requires additional operations on polygons. It has to subtract one polygon from another to find the nonobstructed part of the incident beam cross-section, which propagates beyond the obstructing facet. This operation produces, in general, a nonconvex polygon or a polygon with a hole. As ABT deals with convex polygons, the algorithm is able to perform a suitable partitioning of nonconvex beams. Moreover, during the clipping procedure, the edge portions belonging to the beam cross-section are singled out as they will be considered, in a following step, sources of diffracted beams. On the other hand, a sorting of the facets forming the scene is necessary to avoid an exhaustive research of the facet intercepting the propagating beam in complex scenes with several facets [11].

For each one of the aforementioned operations it is possible to use "ad hoc" algorithms, which are optimized in computational geometry and computer graphics. However, the assembling of a set of standalone, optimized algorithms does not assure the optimum of the resulting algorithm, so in this work we shall focus on the global computational cost.

III. Computational Charge Results

We have executed some numerical experiments increasing the environmental complexity and the number of specular reflections. Four parameters have been chosen to figure out the computational charge, i.e., storage requirement for scene data structure, storage requirement for whole computation, time consumption and number of beams. The test has been run on a PC/Windows computer with 1048 MB of memory and XP1500 AMD Athlon processor.

We have considered a typical urban environment with a grid of rectilinear streets and parallel-piped blocks along the street sides as shown in Fig. 1. Each block can have different size and height. The flat ground has been assumed dielectrically inhomogeneous, so many additional faces have been featured to account for this. An isotropic source is placed at the centre of the scene halfway between the maximum and minimum height of the buildings. Other source positions have been tested, however, the central position corresponds to the worst case, i.e., to the largest number of generated beams. We adopted the total number of reflections undergone by each individual beam as termination rule in beam tracing. Other interactions, as transmission through facets or diffraction have been disregarded in the numerical analysis in order to appreciate the relevance on the computational charge of the test case size and of the number of specular reflections. For each test case we have traced beams until they have suffered eight reflections. Table I reports four test cases with number of buildings, number of polygons in the scene and winged-edge memory requirement. The memory requirement to store input data within the WE structure increases linearly with the number of faces, as expected.

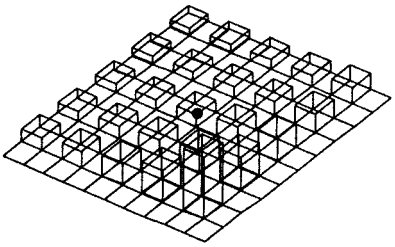

Fig. 1. Model of urban environment and source position in the numerical analysis.

TABLE I
FOUR TEST CASES WITH NUMBER OF BUILDINGS, NUMBER OF POLYGONS IN THE SCENE AND WINGED-EDGE MEMORY REQUIREMENT

Scene	Number of buildings	Number of faces	WE Memory [MB]
S1	9	102	0.029
S2	25	270	0.077
S3	49	518	0.147
S4	81	846	0.240

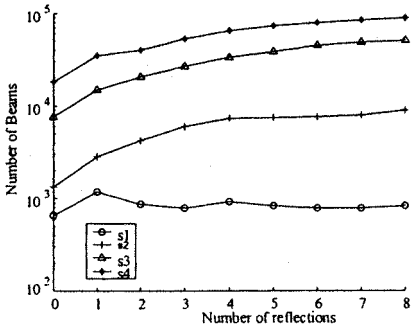

Fig. 2. Number of beams versus reflection order for the four scenes in Table I.

The number of beams is shown in Fig. 2 as a function of the reflection order for the test cases in Table I. The number of reflected beams generally increases with the reflection order owing to the sequential splitting at the faces. The overall number of beams and memory requirement are reported in Fig. 3 for reflection up to the eighth order and increasing scene complexity. The memory is nearly proportional to the number of beams,

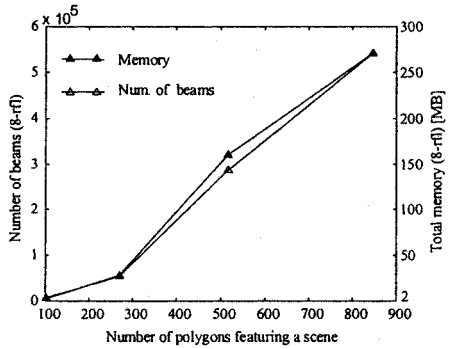

Fig. 3. Total number of beams (left scale) and memory requirement (right scale) versus scene complexity (reflections up to eighth order).

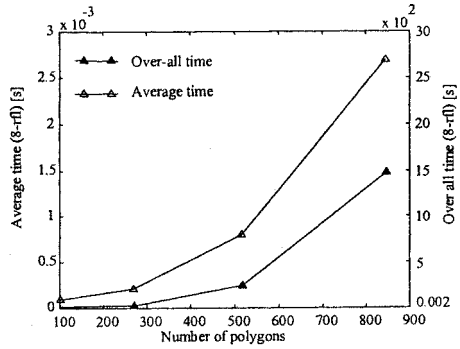

Fig. 4. Average time for single beam tracing (left scale), and total tracing time (right scale) versus scene complexity.

therefore the memory allocated for beam storage prevails over that for WE data. The total time for the visibility test fulfilment is shown in Fig. 4 together with the average time $\bar{t}$ for a single beam tracing. We observe that ABT code has a cost in time, which is well interpolated by $(\alpha n)^4$ with $\alpha = 7.4 \cdot 10^{-3}$ and n number of facets, i.e., the algorithm has a cost proportional to n^4, also indicated as $O(n^4)$. Other ray-tracing algorithms, like direct implementation of forward and backward ray-tracing or of the image method would have a $O(n^8)$ cost for eight reflections.

The average cost per interaction, $\bar{t}$ is calculated as the ratio between the total time to complete the tracing and the overall number of beams. It increases with the number of faces, therefore the code feels the effects of the scene complexity.

IV. Conclusions

ABT has been adapted to a model of urban scenario with increasing complexity and the corresponding computational charge has been evaluated. From the numerical analysis it results that the algorithm generates a large number of beams with execution time increasing with the fourth power of the number of faces in a scene. This feature together with the memory requirement lead to conclude that our code can be applied to the electromagnetic analysis of moderately large scenarios without having recourse to large computational resources. An optimization of the code together with the further use of "ad hoc" algorithms aimed to perform a better partitioning of the scene would improve the performance.

References

[1] M. Barbiroli, C. Carciofi, G. Falciasecca, M. Frullone, and P. Grazioso, "A measurement-based methodology for the determination of validity domains of prediction models in urban environment," *IEEE Trans. Veh. Technol.*, vol. 49, pp. 1508–515, Sept. 2000.

[2] M. F. Catedra, J. Perez, F. S. de Adana, and O. Gutierrez, "Efficient ray-tracing technique for three dimensional analyzes of propagation in mobile communications: application to picocell and microcell scenarios," *IEEE Antennas Propagat. Mag.*, vol. 40, no. , 2, pp. 15–27, Apr. 1998.

[3] J. Maurer, O. Drumm, D. Didascalou, and W. Wiesbeck, "A novel approach in the determination of visible surfaces in 3D vector geometries for ray-optical wave propagation modeling," in *IEEE Veh. Technol. Conf.*, 2000, pp. 1651–1655.

[4] J. Haala and W. Wiesbeck, "Modeling microwave and hybrid heating process including heat radiation effects," *IEEE Trans Microwave Theory Tech.*, vol. 50, no. 5, pp. 1346–1354, 2002.

[5] E. Di Giampaolo, M. Sabbadini, and F. Bardati, "Astigmatic beam tracing for GTD/UTD methods in 3-D complex environments," *J. Electromagn. Waves Applicat.*, vol. 15, no. 4, pp. 439–460, 2001.

[6] E. Di Giampaolo, F. Bardati, and M. Sabbadini, "Astigmatic beam tracing for propagation modeling," in *2001 European Conference on Wireless Technology*, London, U.K., Sept. 27–28, 2001, pp. 273–276.

[7] D. J. Foley, A. van Dam, S. k. Feiner, and J. F. Hughes, *Computer Graphics Principle and Practice.* Reading, MA: Addison-Wesley, 1992.

[8] B. G. Baumgart, *A Polyhedron Representation for Computer Vision*. New York: NCC, 1975, pp. 589–596.

[9] K. Weiler, "Edge-based data structures for solid modeling in curved-surface environments," *IEEE Comput. Graph. Applicat.*, vol. 5, no. 1, pp. 21–40, January 1985.

[10] T. Woo, "A combinatorial analysis of boundary data structure schemata," *IEEE Comput. Graph. Applicat.*, vol. 5, no. 3, pp. 19–27, Mar. 1985.

[11] N. Daudon, D. G. Kirkpatrick, and J. P. Walsh, "The geometry of beam tracing," in *Proc. Symp. Computational Geometry*, June 1985, pp. 55–61.