Progress In Electromagnetics Research, PIER 81, 329–341, 2008

KD-TREE BASED FAST RAY TRACING FOR RCS PREDICTION

Y. B. Tao, H. Lin, and H. J. Bao

State Key Laboratory of CAD&CG
Zhejiang University
Hangzhou 310027, P. R. China

Abstract—Ray tracing is of great use for computational electromagnetics, such as the well-known shooting and bouncing ray (SBR) method. In this paper, the kd-tree data structure, coupled with the mailbox technique, is proposed to accelerate the ray tracing in the SBR. The kd-tree is highly effective in handling the irregularly distribution of patches of the target, while the repeatedly intersection tests between the ray and the patch when using space division acceleration structures can be eliminated through the mailbox technique. Numerical results show excellent agreement with the measured data and the exact solution, and demonstrate that the kd-tree as well as the mailbox technique can greatly reduce the computation time.

1. INTRODUCTION

There are many important applications of ray tracing in computational electromagnetics, such as radio wave propagation [1–7] and the prediction of Radar Cross Section (RCS) [8]. In this paper, we explore a fast and efficient algorithm to accelerate the ray tracing in the shooting and bouncing ray (SBR) [9, 10] method for the RCS prediction of arbitrarily shaped targets.

The SBR is well known for providing more accurate results by including the scattering effect arising from multiple bounces. The incident plane wave in the SBR is described as a dense grid of parallel rays, which are shot toward the target. Each ray is recursively traced according to the law of Geometrical Optics (GO), until it leaves the target, and then Physical Optics (PO) [11–13] is preformed to obtain the scattered field of this ray tube. The total number of initial rays is decided by the electrical size of the target, since the density of incident rays on the virtual aperture perpendicular to the incident ray direction

should be greater than about ten rays per wavelength in view of the convergence. For each ray, the number of ray-patch intersection tests by using the brute force method, is proportional to the number of all patches. Therefore, the ray-tracing procedure in the SBR is very time-consuming for electrically large and complex targets.

Two most popular strategies to accelerate ray tracing are using fewer rays and faster intersection. The adaptive grid division method has been proposed to reduce the initial number of rays [14, 15]. On the other hand, the key idea to reduce the number of intersection tests is the use of acceleration data structures. Once the acceleration structure is constructed, the rays for all incident and reflected rays could be traced effectively. The octree, recursively subdividing the box into eight children boxes using three axis-perpendicular planes, has been used to reduce the number of intersection tests with the assumption that patches are uniformly distributed in the target space [15, 16].

However, this assumption is not always the case, for example, patches of the ship or the aircraft usually have an uneven spatial distribution. It is necessary, therefore, to find an effective acceleration data structure for the uneven spatial distribution of patches. In computer graphics, different acceleration data structures across a variety of scenes (targets) have been widely studied to evaluate their relative performance, and the conclusion is that the kd-tree, which adaptively subdivides the target space into uneven boxes, is the best general-purpose acceleration structure for ray tracing of static scenes [17]. Thus, this paper seeks to make use of the kd-tree to accelerate the ray tracing in the SBR.

One drawback of space division acceleration structures, such as the octree, uniform grid, and kd-tree, is that a ray may be tested for intersection with the same patch multiple times, since the patch may overlap multiple nodes of the acceleration structure. These redundant intersection tests can be avoided using the mailbox technique [18], whose implementation is explained in subsection 2.2.

In summary, the kd-tree acceleration structure in conjunction with the mailbox technique is proposed for the ray tracing in the SBR in this paper. The proposed method combined with the adaptive grid method can significantly accelerate the ray tracing in the RCS prediction for electrically large and complex targets.

2. RAY TRACING USING THE KD-TREE ACCELERATION STRUCTURE

The kd-tree, which is a variation of the binary space partitioning tree, recursively employs the axis-perpendicular plane to split the target

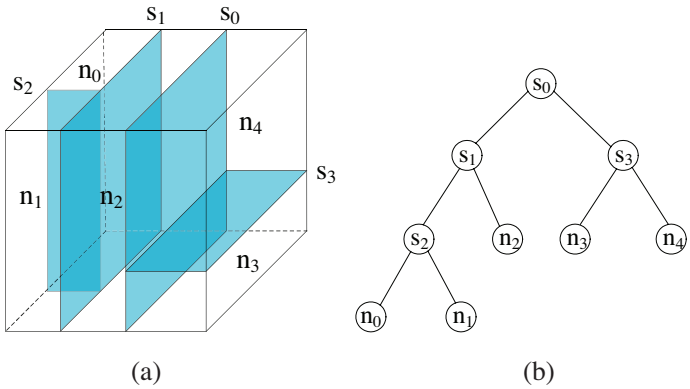


Figure 1. (a) A 3D kd-tree. Interior nodes are labeled as their splitting planes and leaf nodes are labeled in their boxes. (b) A graph representation of the same kd-tree.

space into uneven axis-aligned boxes, as illustrated in Fig. 1. The typical construction and traversal procedure, taken from Havran's thesis [17] as well as Pharr and Humphreys' book [19], are reviewed as follows, and how to apply the kd-tree in the SBR is also described.

2.1. Kd-Tree Construction

The kd-tree is constructed recursively from top to bottom, and the root node corresponds to the bounding box of the target and contains all patches. At each step of the recursive construction, a node, which contains a group of patches that overlap the axis-aligned box of the node, is processed to be an interior node or a leaf node. If the number of patches in this node is less than the user-defined number, or the depth of this node is above the maximum depth, or there is no benefit to split this node, a leaf node will be created with its associated patches, and the recursion of this node is terminated. Otherwise, this node will be split in half by an axis-perpendicular plane and become an interior node. Patches of this node are then associated with the child node they overlap, and if the patch is across the splitting plane, it should be associated with both children nodes. These two children nodes are then processed recursively until the termination condition is satisfied.

In the interior node, the position of the splitting plane can be placed at arbitrary position of three axes inside the node. The choice of the splitting position is based on the ray-tracing cost estimation model, in which the cost includes the traversal time of interior nodes

and the ray-patch intersection time of leaf nodes. The optimal splitting position is the one that adapts to the geometry of patches and minimizes the total ray-tracing cost. This is the most notable difference between the octree and the kd-tree, as at each step the octree simply splits the box using three axis-perpendicular planes at the middle point of the extent in each direction. This is also why the kd-tree can provide faster ray tracing for the irregularly distribution of patches in the target space.

The best known heuristic is the greedy Surface Area Heuristic (SAH) that minimizes the estimated cost for the node individually to construct the approximately optimal kd-tree based on the assumption of uniformly distributed rays [20]. The assumption is reasonable in the SBR, since it would be possible to predict the RCS of the target in any incident direction and the incident and reflected rays would be uniformly distributed in all directions. Under this assumption, the geometric probability states that if a ray is known to pass through an interior node N, the conditional probability of hitting the child node N' is the ratio of their surface areas, $S_{N'}$ and S_N:

$$P(N'|N) = S_{N'}/S_N. \tag{1}$$

Therefore, when a random ray is intersected with an interior node N, the estimated cost is composed of the traversal cost and the probability-weighted ray-patch intersection costs of its children nodes N_l and N_r. The traversal cost C_t is the time of traversing the interior node and determining the traversal order of children nodes. The ray-patch intersection cost of a child node with n associated patches is simplified to n times one ray-patch intersection cost C_i. The traversal cost C_t and one ray-patch intersection cost C_i are relative numerical values defined by the user. Thus, the estimated cost C_N for an interior node N is

$$C_N = C_t + [n_l C_i P(N_l|N) + n_r C_i P(N_r|N)], \tag{2}$$

where n_l and n_r are the number of patches contained in children nodes N_l and N_r, respectively.

It is clear that the optimal splitting position is the one where the estimated cost C_N is minimal, which can only be obtained at a split position that is coincident with one of the planes of the patch's bounding box. As a result, split candidates are the bounding planes of patches inside the node. Fig. 2 shows an example of split candidates along the x-axis. For each split candidate, the surface area of children nodes can be directly computed. However, the fast evaluation of the patch numbers needs a carefully structured algorithm considering the

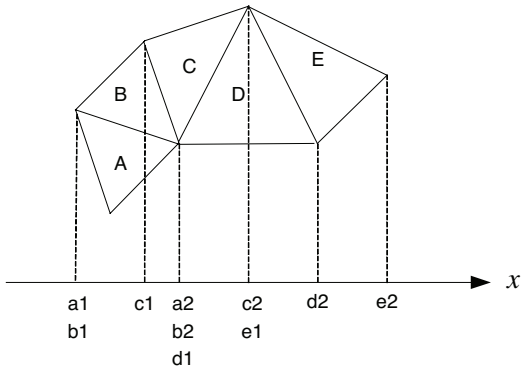


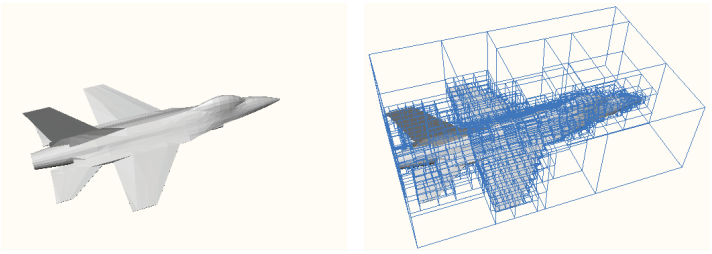
Figure 2. Split candidates along the x-axis. Each patch has two split candidates, for example, a1 and a2 for patch A.

efficiency. The process to search the optimal split position of one axis is described as follows:

1) The split candidates are initialized by projecting the bounding boxes of associated patches onto this axis. Each bounding box has two split candidates, and each candidate contains the position on this axis, the flag that represents the start or end of a bounding box, and the patch it belongs to.
2) Sort these split candidates from low to high along the axis.
3) Sweep across the sorted split candidates iteratively, incrementally update the patch number of children nodes, and compute the estimated cost of each split candidate.
4) Output the position of the split candidate with the minimal cost.

The optimal split position and dimension is the one with the minimal cost among three coordinate axes. It is possible that the direct intersection cost nC_i is less than the minimal cost C_N, where n is the number of patches associated in this node. If this happens, a leaf node will be created, because there is no benefit to split this node. Otherwise, split this node with the minimal-cost splitting plane, then associate patches of this node with children nodes, and continue the recursion construction.

The above description is the basic recursive construction, and further optimizations can be found in [19]. As sorting is used to find the optimal splitting plane at each step, the expected complexity of this recursive construction is $O(N \log^2 N)$, where N is the number of all patches. Fig. 3 shows an aircraft and the splitting planes of its



(a) (b)

Figure 3. (a) An aircraft model. (b) The splitting planes of the kd-tree of the same aircraft model.

kd-tree, and it can be noticeable that the target space is subdivided into axis-aligned boxes with different sizes.

2.2. Kd-Tree Traversal with the Mailbox Technique

The ray tracing in the SBR searches for the nearest patch of the target that is intersected with a geometrical-optics ray. At each intersection position, GO is applied to compute the reflected direction and reflected field, and recursively trace the reflected ray, until the ray exits the target. The kd-tree traversal algorithm can be used to reduce the time to find the nearest intersection between the ray and the target.

The traversal starts with the root node of the kd-tree, and a stack is used as a priority queue of nodes left to visit according to how closer to the ray origin. A $(t_{\min}, t_{\max})$ range defines the part of the ray that is inside the current node.

At each interior node, its children nodes can be classified into the near and far node in the ray-traversal order according to the relative position between the ray's origin and the splitting plane. The distance t_{split} from the ray origin to the splitting plane along the ray may allow us only need to traverse one child node. The near node only needs to be traversed, if the ray faces away from the far node ($t_{split} < 0$) or the range $(t_{\min}, t_{\max})$ lies entirely in the near node ($t_{split} > t_{\max}$), as illustrated in Fig. 4(a) and Fig. 4(b), respectively. If the ray intersects the splitting plane before the intersection position between the ray and the node ($t_{\min} > t_{split}$), as showed in Fig. 4(c), only the far node needs to be traversed. Otherwise, the both nodes need to be processed, traversal firstly continue to the near node, and the far node is pushed onto the stack together with its appropriate $(t_{\min}, t_{\max})$ range. Following the above rules, the kd-tree is recursively traversed down until a leaf node is encountered.

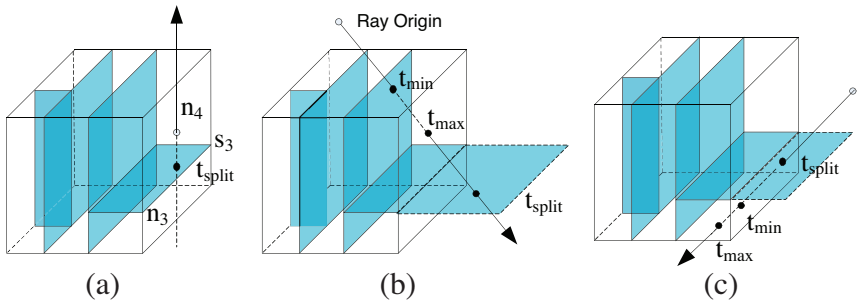


Figure 4. Ray traversal of the interior node s_3 (Fig. 1). According to the origin position with respect to the splitting plane, the near and far nodes are n_4 and n_3, respectively. Only the near node n_4 needs to be traversed in cases (a) and (b), and traversal only need to continue to the far node n_3 in case (c).

At each leaf node, the ray is iteratively tested for intersection with patches of the leaf node to find the nearest intersection. As the patch may be referenced in multiple leaf nodes and these leaf nodes are usually neighboring nodes, there is a high probability that a ray may be tested with the same patch multiple times. These redundant tests can be avoided through the mailbox technique [17–19]. In this technique, a unique integer index is assigned to each ray, and each patch also keeps the index of the last tested ray. Before the ray is tested for intersection with a patch, the index of the ray is compared with the index of the patch. If they have the same index, then the ray-patch intersection test has already been performed for this pair and it is unnecessary to compute again. If the patch's index is different from the index of the current ray, the ray-patch intersection test is performed for this pair, and the index of the patch is updated to be the index of the current ray. In this manner, the mailbox technique is guaranteed that the number of ray-patch intersection tests using the kd-tree will never be greater than the brute force method.

If the ray intersected one of patches and the intersection is inside the leaf node (the distance along the ray is within the $(t_{\min}, t_{\max})$ range), the nearest intersection is found. At the intersection position, the reflected ray is generated by GO and continues to be traced. Otherwise, the next node with its $(t_{\min}, t_{\max})$ range is popped from the stack and the traversal continues. If the stack is empty, the ray passes through the target space without intersection, and then PO is applied to calculate the far field contribution of this ray tube.

The expected complexity of this traversal algorithm for general

targets with N leaf nodes is $O(\log N)$, which is usually proportional to the height of the kd-tree.

3. NUMERICAL RESULTS AND DISCUSSION

The RCS of several targets are studied to validate the accuracy of the ray tracing in the SBR using the kd-tree acceleration structure combined with the mailbox technique. Additionally, the adaptive grid method is applied simultaneously for following examples. Although the initial size of the ray tube should be large enough to speed up the prediction, it should be small enough to capture the tiny feature of the target and to keep the same accuracy. Hence, the initial size and the minimum size are defined as 1 and 0.0625 wavelength, respectively.

The first simple target is a right-angle dihedral corner with the width 133 mm and the height 138 mm as illustrated in Fig. 5(a), which has been studied in [14]. The monostatic RCS result for vv-polarization is shown in Fig. 5(b) with an angular resolution of 1° at 10 GHz frequency. The result is in good agreement with the measured data [14].

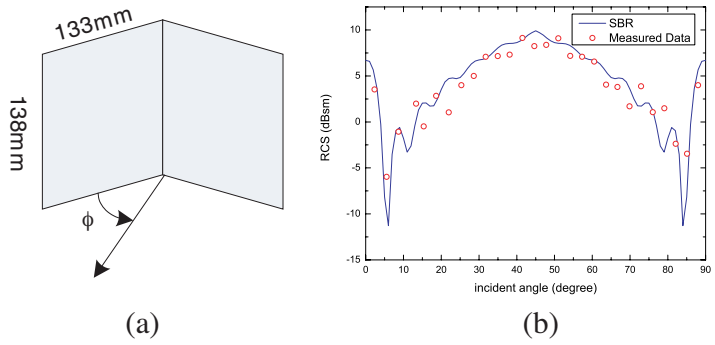


Figure 5. (a) A right-angle dihedral corner. (b) Comparison of our result and measured data [14] for the corner at 10 GHz, vv-polarization.

The simple ship is the second validated target defined in [14–16], where some of the geometric details are unknown. Hence, a new ship is modeled based on the available geometric parameters ($0.9\times0.2\times0.2$ m), as shown in Fig. 6(a). As this ship is not identical with the one in [14–16], which provide the measured data, a MLFMM result is used to verify the accuracy. Fig. 6(b) shows the monostatic RCS comparison of hh-polarization of the SBR result and the MLFMM result with an angular resolution of 1° at 10 GHz frequency. A good agreement is

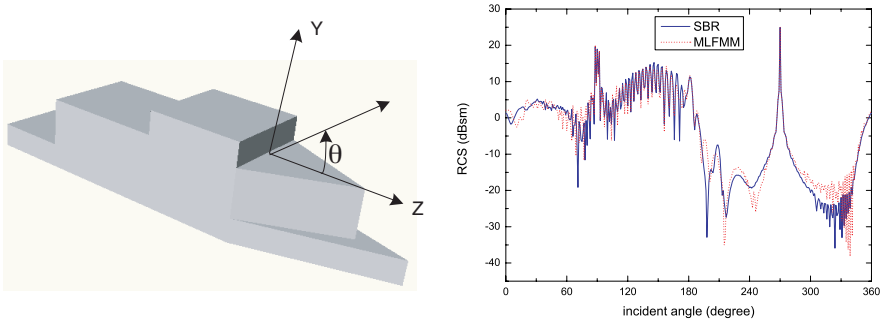


Figure 6. (a) The geometry of the simple ship. (b) Comparison of our result and the MLFMM result for the ship at 10 GHz, *hh*-polarization.

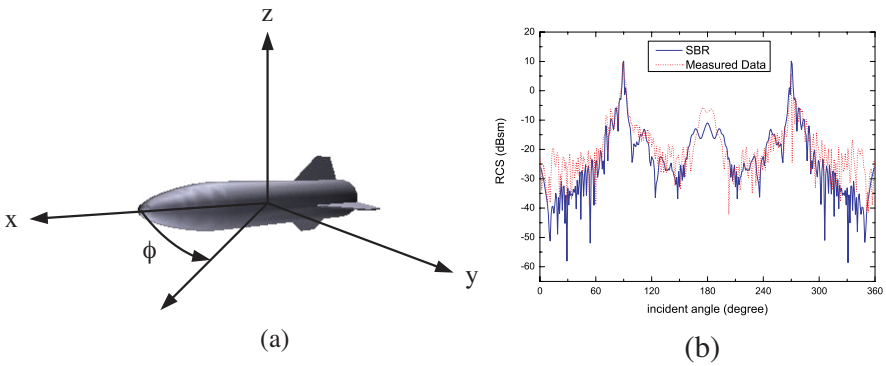


Figure 7. (a) The geometry of generic complex missile. (b) Comparison of our result and measured data of the missile at 7.5 GHz, *hh*-polarization.

observed between these two results.

The final target is the generic complex missile. Fig. 7 illustrates the geometry of the generic missile ($1.1 \times 0.25 \times 0.2$ m) and the *hh*-polarization result compared with the measured data. The monostatic RCS is predicted using an angular resolution of 1° at 7.5 GHz frequency and the fifth-order reflection is maximum order of reflection. The results show an acceptable agreement, and the deviation may be due to the edge-diffraction effect, which is not included in the prediction code.

The aim of this paper is the use of the kd-tree and the mailbox technique to accelerate the ray tracing in the SBR, therefore, an

elaborate study on the computational efficiency is presented in the fashion similar to [14–16]. All calculations were performed in a 2.8 GHz Pentium(R) D CPU and the computation times are the total time of all incident angles. The adaptive grid method is also applied for all methods using the same parameters. Two different types of targets are investigated for different acceleration strategies. Each target is described with a different number of patches to analysis the relationship between the computation time and the number of patches. The height of the octree is five the same as [15, 16].

The first target is the simple ship in Fig. 6. Table 1 shows the change in the computation time of the SBR with the number of patches using the brute force method, the octree-based method, and the proposed method. As can be seen from Table 1, the computation time is reduced significantly compared with the brute force method, at least 6 times faster, and even for the small number of patches, the time using the proposed method is smaller than the one accelerated using the octree.

Table 1. The computation time of the simple ship (sec).

Numbers of patches	484	730	1286	1792	2440
Brute force method	345.86	600.26	1270.09	1911.59	2951.62
Octree-based method	92.11	108.93	140.05	161.62	186.61
Proposed method	54.22	62.95	83.21	93.99	109.53

The other target is the aircraft with complex structures ($11.76 \times 7.4 \times 3.67$ m) in Fig. 3. The calculation parameters of the airplane are the same as the one of the missile in Fig. 7, except that the operating frequency is 10 GHz. Fig. 8 shows the computation time of the SBR using the octree or kd-tree with/without the mailbox technique. The numbers of patches evaluated in this example are 5 250, 8 724, 13 050, 16 988, and 22 294, respectively. As observed in Fig. 8, the mailbox technique not only reduces the time of ray tracing using the kd-tree, but also achieves an approximate 5% speedup for the octree. This observation confirms that the mailbox technique can avoid the redundant tests for space division acceleration structures, and subsequently the time of ray tracing is reduced. As the number of patches increases, the kd-tree taking into account the uneven distribution of patches is at least 2 times faster than the octree for the ray tracing in the SBR. The kd-tree scales well with the number of patches, as when the number of patches increase from 5 250 to 22 294,

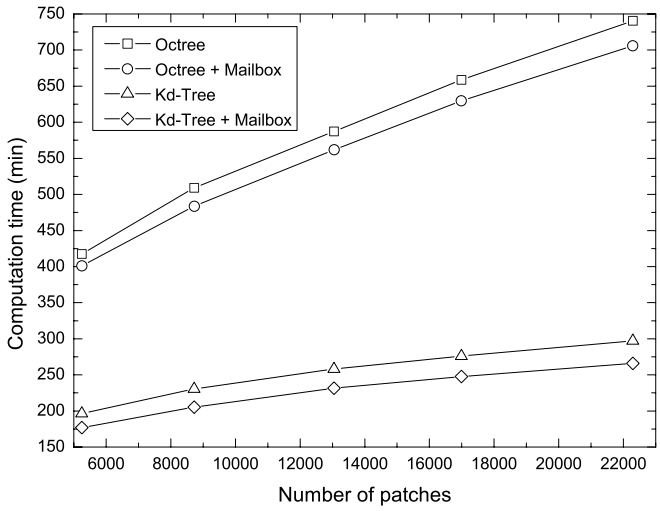


Figure 8. Computation time of the SBR method using the octree and kd-tree with/without the mailbox technique.

the computation time increases only about 50% as shown from Fig. 8, and the memory requirement of the kd-tree for the largest number of patches is only about 15 M.

4. CONCLUSION

It has been shown that the kd-tree acceleration structure constructed based on the distribution of patches, significantly improves the computational efficiency for the ray tracing in the SBR, while the mailbox technique is used to prevent the repeatedly intersection with the same patch in space division acceleration structures. Numerical results demonstrate the accuracy and efficiency for the RCS prediction. It can be expected that the kd-tree acceleration structure and the mailbox technique can also be adapted to other computational electromagnetic methods that use the ray tracing.

ACKNOWLEDGMENT

The authors would like to thank Professor Tiejun Cui from South East University for providing their MLFMM method used in this paper. This paper is supported by China 863 Hi-Tech Research and Development Program (2002AA135020) and China 973 Program (2002CB312102).

REFERENCES

1. Fügen, T., J. Maurer, T. Kayser, and W. Wiesbeck, "Capability of 3-D ray tracing for defining parameter sets for the specification of future mobile communications systems," *IEEE Transaction on Antena and Propagation*, Vol. 54, No. 11, 3125–3137, 2006.
2. Wang, S., H. B. Lim, and E. P. Li, "An efficient ray-tracing method for analysis and design of electromagnetic shielded room systems," *Journal of Electromagnetic Waves and Application*, Vol. 19, No. 15, 2059–2071, 2005.
3. Chen, C. H., C. L. Liu, C. C. Chiu, and T. M. Hu, "Ultra-wide band channel calculation by SBR/Image techniques for indoor communication," *Journal of Electromagnetic Waves and Application*, Vol. 20, No. 1, 41–51, 2006.
4. Teh, C. H., F. Kung, and H. T. Chuah, "A path-corrected wall model for ray-tracing propagation modeling," *Journal of Electromagnetic Waves and Application*, Vol. 20, No. 2, 207–214, 2006.
5. Mphale, K. and M. Heron, "Ray tracing radio waves in wildfire envionment," *Progress In Electromagnetics Research*, PIER 67, 153–172, 2007.
6. Martini, A., L. Marchi, M. Franceschetti, and A. Massa, "Stochastic ray propagation in stratified random lattices — comparative assessment of two mathematical approaches," *Progress In Electromagnetics Research*, PIER 71, 159–172, 2007.
7. Cocheril, Y. and R. Vauzelle, "A new ray-tracing based wave propagation model including rough surfaces scattering," *Progress In Electromagnetics Research*, PIER 75, 357–381, 2007.
8. Weinmann, F., "Ray tracing with PO/PTD for RCS modeling of large complex objects," *IEEE Transaction on Antena and Propagation*, Vol. 54, No. 6, 1797–1806, 2006.
9. Ling, H., R. C. Chou, and S. W. Lee, "Shooting and bouncing rays: calculating the RCS of an arbitrarily shaped cavity," *IEEE Transaction on Antena and Propagation*, Vol. 37, No. 2, 194–205, 1989.
10. Baldauf, J., S. W. Lee, L. Lin, S. K. Jeng, S. M. Scarborough, and C. L. Yu, "High frequency scattering from trihedral corner reflectors and other benchmark targets: SBR vs. experiments," *IEEE Transaction on Antena and Propagation*, Vol. 39, No. 9, 1345–1351, 1991.

11. Chen, M., Y. Zhang, and C. H. Liang, "Calculation of the field distribution near electrically large NURBS surfaces with physical optics method," *Journal of Electromagnetic Waves and Application*, Vol. 19, No. 11, 1511–1524, 2005.
12. Zhang, P. F. and S. X. Gong, "Improvement on the forwardbackward iterative physical optics algorithm applied to compute the RCS of large open-ended cavities," *Journal of Electromagnetic Waves and Application*, Vol. 21, No. 4, 457–469, 2007.
13. Zhong, X. J., T. J. Cui, Z. Li, Y. B. Tao, and H. Lin, "Terahertz-wave scattering by perfectly electrical conducting objects," *Journal of Electromagnetic Waves and Application*, Vol. 21, No. 15, 2331–2340, 2007.
14. Suk, S. H., T. I. Seo, H. S. Park, and H. T. Kim, "Multiresolution grid algorithm in the SBR and its application to the RCS calculation," *Microwave and Optical Technology Letters*, Vol. 29, No. 6, 394–397, 2001.
15. Bang, J. K., B. C. Kim, S. H. Suk, K. S. Jin, and H. T. Kim, "Time consumption reduction of ray tracing for RCS prediction using efficient grid division and space division algorithms," *Journal of Electromagnetic Waves and Application*, Vol. 21, No. 6, 829–840, 2007.
16. Jin, K. S., T. I. Suh, S. H. Suk, B. C. Kim, and H. T. Kim, "Fast ray tracing using a space-division algorithm for RCS prediction," *Journal of Electromagnetic Waves and Application*, Vol. 20, No. 1, 119–126, 2006.
17. Havran, V., "Heuristic ray shooting algorithms," Ph.D. thesis, Faculty of Electrical Engineering, Czech Technical University in Prague, 2000.
18. Kirk, D. and J. Arvo, "Improved ray tagging for voxel-based ray tracing," *Graphics Gems II*, 264–266, Academic Press, 1991.
19. Pharr, M. and G. Humphreys, *Physically Based Rendering: From Theory to Implementation*, Morgan Kaufmann, Har/Cdr edition, 2004.
20. Macdonald, J. D. and K. S. Booth, "Heuristics for ray tracing using space subdivision," *In Proceedings of Graphics Interface*, 152–163, 1989.