A Discrete Radiosity Method

Rémy Malgouyres

A Discrete Radiosity Method

Rémy Malgouyres

LLAIC, Université Clermont 1, IUT departement Informatique, B.P. 86, 63172 AUBIERE cedex, France. e-mail: Remy.Malgouyres@llaic.u-clermont1.fr

Abstract. We present a completely new principle of computation of radiosity values in a 3D scene. The method is based on a voxel approximation of the objects, and all occlusion calculations involve only integer arithmetics operation. The method is proved to converge. Some experimental results are presented.

Key words: Discrete Graphical Models, Voxel, Global Illumination, Radiosity.

Introduction

Radiosity ([SP94]) is a technique which has proved being efficient (in spite of its large complexity) and accurate to simulate several physical processes involving exchange of energy. Its fields of application include weather forecast, heat transfer simulation and light propagation, and in particular 3D rendering in computer graphics.

However, radiosity remains so expensive that it cannot yet be extensively used in the movie industry and similar applications. Moreover, the basic radiosity algorithms rely on the so-called ideal diffuse hypothesis, which represents a limitation of the range of applications, and often implies a complicated combination of radiosity with other techniques (such as ray-tracing) in order to obtain a correct simulation.

On the other hand, integer only geometric computations are everyday improved and several techniques have been introduced for the use of discrete geometric models in the field of modeling or computer graphics ([YCK92], [SC95], [ANF97], [BM00]). Though discrete ray-tracing has not become very widely used, it remains an interesting discrete geometric modelisation and a good first attempt to use discrete geometric models for 3D rendering.

The purpose of this paper is to introduce a completely new radiosity method, which is based on a voxel representation of objects, and whose occlusion calculations involve only integer arithmetical operations. The

objects of the scene are first approximated by a set of voxels, which are stored in an octree data structure. Then the radiosity computations are performed in the discrete voxel space. Finally, some classical techniques such as $z-$buffer or ray-tracing are used for display of the results.

The paper is organized as follows: first we remind the reader the required notions about classical radiosity, then we set the basics of discretization. Afterwards, we explain how we can approximate the so-called diffuse illumination equation by a discrete equation, and propose a numerical solution which is proved to converge towards a solution of our discrete equation. Then comes a section about implementation and complexity of the method, and finally we present some experimental results.

1 Basic Notions for Radiosity

In this section we recall the basics of the radiosity method. We refer to [SP94] for more details on classical radiosity methods.

1.1 The data of a 3D scene

We assume that we are given a *3D scene*, which consists of a set of polyhedra (also called *objects*), or surfaces approximated by polyhedra, each polyhedron P of the scene being provided with a *reflectance coefficient* $\rho_P \in [0, 1[$. The physical definition of this reflectance coefficient is intuitively that ρ_P is the ratio of the total power of the outgoing light at a given point of the polyhedron and of the power of the incoming light at the same point. Note that the hypothesis that $\rho_P < 1$ means that no energy is created while light is reflected by the object. For convenience, given a point x of the object P, we denote $\rho(x) = \rho_P$.

In this paper, we adopt the simplifying hypothesis that the reflectance ρ_P does not depend on the point on the polyhedron P, which means that the objects are made of a uniform material: they are not *textured*. This is not an intrinsic limitation of the presented method, but rather a hypothesis that we made for the first investigations.

There is also another hypothesis, which is that the reflectance does not depend on the incoming and outgoing direction of the light. This hypothesis is known as the *ideal diffuse reflectors* hypothesis, and is classically the basic hypothesis in radiosity methods. Intuitively, an ideal reflector is an object such that the light emitted from a point x of this object has the same properties in all the directions of the half-space limited by the tangent plane at point x and which contains the outgoing normal vector at x.

A *light source* is a particular object P which is provided with a positive real number E_P called the *exitance* of P. Intuitively, the exitance is the total power of light leaving the object per unit of area. As the reflectance, the exitance is assumed to depend neither on the point of the object, nor on the direction of emission. Given a point x of the object P, we denote $E(x) = E_P$.

1.2 The Diffuse Illumination Equation

Here we describe a continuous equation which expresses the power of light leaving a point x (*radiosity* at point x) of an object P as a function of the power of light of all other points in the scene. Let us denote by $B(x)$ the radiosity at point x. Given a point y in the scene, we denote by $V(x, y)$ the number equal to 1 if y is visible form x in the scene (i.e. if no object of the scene intersects the straight line segment $[x, y]$), and equal to 0 otherwise. The function V thus defined is called the *visibility function*. We denote by $\theta(x, y)$ the angle between the vector $\overrightarrow{xy}$ and the normal vector $\overrightarrow{n}$ at point x if this angle is less than $\frac{\pi}{2}$, and equal to $\frac{\pi}{2}$ otherwise (so that $\cos\theta(x, y) = 0$ if y is not in the half space limited by the tangent plane at x and containing the outgoing normal vector).

The *diffuse illumination equation* at point x, which corresponds to the physical ideal diffuse reflection model, is the following (see [SP94]):

$$B(x) = E(x) \; + \; \rho(x) \int_{y \in scene} B(y) \; \frac{\cos\theta(x, y) \;\; \cos\theta(y, x)}{\pi \; r^2} \; V(x, y) \; dy \quad (1)$$

where r denotes the distance between x and y.

The problem of simulating light propagation and reflections in the scene, which consists in computing the radiosity $B(x)$ at each point x of the scene, reduces to solving the diffuse illumination equation. However, this integral equation cannot be solved analytically except in some very particular cases which are of no use in the field of computer graphics. Therefore, we have to find an approximate numerical solution.

A classical way to do this, called the *radiosity method*, is do break down the objects of the scene into a finite number of patches, and solving a discrete version of the diffuse illumination equation, the solution of this discrete version of the equation consisting in solving a (huge) matrix equation. However, in this solution, the coefficients of the involved matrix depend on the so-called *form factors*, which are double integrals over the surface of couples of patches. The computation of these form factors constitute the main part of the runtime of radiosity programs, and is either very slow, or quite rough.

The idea of this paper is to discretize the scene to obtain voxels, and, roughly speaking, to consider each voxel as a single point in order to replace the computation of integrals by computation of sums.

2 Discretization of a 3D scene

2.1 Basic Notions of Discrete Geometry

A *voxel* $v = (i, j, k)$ is a point of $\mathbb{Z}^3$, i.e. a point with integer coordinates. Classically, such a voxel $v = (i, j, k)$ can be seen as a unit cube centered at the point (i, j, k), and whose edges are parallel to coordinates axis.

Given two voxels $v = (i, j, k)$ and $v' = (i', j', k')$, we say that v and v' are $26-$adjacent if $\max(|i - i'|, |j - j'|, |k - k'|) = 1$. We say that v and v' are $18-$adjacent if they are $26-$adjacent and have at least one coordinate in common. Finally, v and v' are said to be $6-$adjacent if they are $26-$adjacent and differ only by one of their coordinates. Given v a voxel and $n \in \{6, 18, 26\}$, we call $n-$neighborhood of v, and we denote by $N_n(v)$ the set of all voxels which are $n-$adjacent to v.

Let $n \in \{6, 18, 26\}$. Using the notion of $n-$adjacency, we can define $n-$connectivity as follows. Let $X \subset \mathbb{Z}^3$ be a set of voxels. An $n-$path in X form v_0 to v_p is a finite sequence $(v_0, \ldots, v_p)$ of elements of X such that for $i = 1, \ldots, p$ the voxels v_i is $n-$adjacent to v_{i-1}. Now let v and v' be two elements of x. We say that v and v' are $n-$*connected in* X if there exists an $n-$path in X from v to v'. The relation "to be $n-$connected in X" is an equivalence relation, and we call $n-$*connected components of* X the equivalence classes of this equivalence relation. The set X is called $n-$*connected* if it has exactly one $n-$connected component.

Given $X \subset \mathbb{Z}^3$, we denote by $\overline{X}$ the complement $\mathbb{Z}^3 \backslash X$ of X in $\mathbb{Z}^3$. The set X is said to be $n-$*separating* if $\overline{X}$ has exactly two $n-$connected components.

2.2 Discretizing a polyhedron

The main problem is the following: given a closed polyhedron P, with an interior and an exterior, how to generate a list of voxels which approximate the polyhedron P (in the sense for instance that the Hausdorf distance between the obtained set of voxels and P is less than 1) , and such that the obtained set of voxels is $6-$separating. There is no solution to this problem if P is an arbitrary polyhedron (just think of pinched, thin or highly curved polyhedra), but, as we shall see from experimental results, we can find practically acceptable solutions for thick enough

polyhedra. We use a discretization scheme similar to the one described in [BM02]: for each face of the polyhedron, we use a polygon filling algorithm ([FVD96]) to go over the pixels of the projection of the face onto (say) the $z = 0$ plane, and for each of these pixels we sample the height z of the face over this pixel.

2.3 Our Discrete Data-Structure

In order to represent the discrete scene obtained after discretization, we have chosen an octree data structure, for the following reasons. First it is much less memory consuming than a boolean matrix representation of discrete objects, and second it is compatible with a hierachical version of the method, i.e. a version of the method in which some parts of the scene are discretized more roughly than others. As we shall see in the conclusion, such a hierachical method is the only way to obtain a technique which can seriously be compared to the most recent and advanced versions of classical radiosity methods.

Since octrees have been very widely used in computer graphics, and more generally in computer imagery since long ago (see [S85] and [W95] among many others), we do not describe them here into many details. We just mention that an octree is a tree structure in which each node has less than 9 children, each child of a node representing an eighth portion of the space represented by the node. The root of the tree thus represents the whole matrix. A node is a leaf of the tree either if the portion of space it represents is only composed of 1's (object leaf case), or if this portion of space is only composed of 0's (complement leaf case). Thus, it is possible to represent an $n \times n \times n$ boolean matrix by a tree with depth (at most) $\log n$. Then, determining if a given element of the matrix is 1 can be done in $O(\log n)$.

In our case, since 1's correspond to voxels approximating surfaces, we are likely to find many wide portions of space composed only of 0's, and thus many low depth complement leaves. This is another argument for the use of the octree data structure. Indeed, as was previously pointed out by authors using octrees for ray-tracing or discrete ray-tracing ([SC95], [YCK92]), and as we shall see below, the existence of low depth complement leaves enables us to speed up the computation of the intersection between a ray (i.e. a half-line) and the objects of the scene.

3 Discretizing and Solving the Diffuse Illumination Equation

3.1 Transforming the Integral Equation

Now we are going to explain how, by transforming a bit the diffuse illumination equation (Equation 1), and by using sums to approximate integrals, we can obtain a discrete equation which can be numerically solved. The first term on the right side of the diffuse illumination equation, due to emitance, can easily be handled, henceforth we concentrate on the second term, which is an integral representing the light leaving the point x which is due to reflection of light arriving from other points y of the scene. We denote by $S_R(x)$ the sphere centered at the point x with radius R, with $R \in \mathbb{R}^+$. Moreover, given $\sigma \in S_R(x)$, we denote by $y(x, \sigma)$ the first point of the scene met when going over the ray (i.e. half line) having x as extremity and containing σ.

$$\begin{aligned} & \int_{y \in scene} B(y) \ \frac{\cos\theta(x,y) \ \cos\theta(y,x)}{\pi \ r^2} \ V(x,y) \ dy \\ = & \int_{\sigma \in S_R} B(y(x,\sigma)) \cos\theta(x,y) \left(\cos\theta(y,x) \frac{d(y(x,\sigma))}{\pi ||x - y(x,\sigma)||^2} \right) \\ = & \int_{\sigma \in S_R} B(y(x,\sigma)) \cos\theta(x,\sigma) \left(\frac{d\sigma}{\pi R^2} \right) \end{aligned}$$

Note that we can identify the terms $\cos\theta(y,x)\frac{d(y(x,\sigma))}{\pi||x-y(x,\sigma)||^2}$ and $\frac{d\sigma}{\pi R^2}$ because both can be recognized as an element of *solid angle* viewed from the point x.

3.2 The Discrete Sphere Method

Now we explain how we approximate the latter integral over a sphere with radius R by a sum over the voxels of a discrete sphere. The idea is simply that the integral of a function can be approximated by sum over small patches of the area of the patch multiplied by the value of the function at some point of the patch. We use a similar idea to the so-called Hemicube method (see [SP94] for instance) in order to discretize the set of directions in space.

We consider a discrete sphere Σ_R with radius R, with $R \in \mathbb{R}$, centered at a voxel $x = (a, b, c)$, i.e. the set of all voxels at distance less than or equal to R from x and having a 6−neighbor at distance more than R from x. In our method, we shall construct as an initialization the voxels of such a discrete sphere. We can use a straightforward construction algorithm, since the time for constructing the sphere will anyway be very small as compared to the overall radiosity method runtime.

Then we consider, for each voxel $v \in \Sigma_R$, the set $F(v)$ of all the faces of the voxel v which are shared by a voxel with distance greater than R from x. All these faces constitute the frontier of the discrete ball with radius R. For each face $f \in F(v)$, we consider the solid angle $A(f)$ formed by f viewed from x. The solid angle $A(f)$ can be approximated as follows: assume for instance that f is a face shared by $v = (a+i, b+j, c+k)$ and the voxel $v' = (a+i, b+j, c+k+1)$. Then

$$A(f) \simeq \frac{k+0.5}{\pi * (i^2 + j^2 + (k+0.5)^2)^{\frac{3}{2}}}.$$

Now, consider S_R the continuous sphere centered at x with radius R over which we want to compute the integral as above. Consider, for each $v \in \Sigma_R$ and each $f \in F(v)$, the patch $p(f)$ which is the central projection (with center x) of f on the sphere S_R. We have:

$$\begin{array}{l} \int_{\sigma \in S_R} B(y(x,\sigma)) \cos\theta(x,\sigma) \left(\frac{d\sigma}{\pi R^2}\right) \\ = \sum_{v \in \Sigma_R, f \in F(v)} \int_{\sigma \in p(f)} B(y(x,\sigma)) \cos\theta(x,\sigma) \left(\frac{d\sigma}{\pi R^2}\right) \\ \simeq \sum_{v \in \Sigma_R, f \in F(v)} B(y(x,v)) \cos\theta(x,v) A(f) \end{array}$$

The latter approximation is obtained by considering the integrand $B(y(x,\sigma)) \cos\theta$ as constant on the patch $p(f)$; the obtained integral is $B(y(x,v)) \cos\theta$ multiplied by the solid angle of $p(f)$ viewed from x, this solid angle being equal to $A(f)$. Now, $y(x,v)$ can be approximated by the first voxel $I(x,v)$ encountered by going over a discrete line from the voxel x through the voxel v. Finally, the diffuse illumination equation of Section 1.2 is approximated by the following discrete linear equation for each voxel x of the discrete scene:

$$B(x) = E(x) \ + \ \rho(x) \sum_{v \in \Sigma_R, f \in F(v)} B(I(x,v)) \cos\theta(x,v) A(f) \qquad (2)$$

3.3 Numerical Solution of the Discrete Equation

The Proposed Algorithm

Lemma 1. *We have* $\lim_{R \to +\infty} \sum_{v \in \Sigma_R, f \in F(v)} \cos\theta(x,v) A(f) = 1.$

An immediate consequence of this lemma is that our linear system (Equation 2) satisfies the formal properties under which the Jacobi relaxation and Gauss-Seidel relaxation both converge to a solution of the

system (see [SP94] for the similar use of Gauss-Seidel relaxation in classical radiosity).

Now we explain the iterative scheme. We set $B_0(x) = E(x)$ for each voxel x of the scene. Then we inductively define $B_i(x)$ for $i \geq 1$ by:

$$B_i(x) = E(x) \ + \ \rho(x) \sum_{v \in \Sigma_R, f \in F(v)} B_{i-1}(I(x,v)) \cos\theta(x,v) A(f) \qquad (3)$$

When i tends to infinity, the numbers B_i for all voxels exponencially converge to a solution of the discrete Equation 2.

4 Implementation and Complexity

4.1 Going Over a Discrete Line

We remind the reader that the voxels of the surfaces are encoded in an octree. The problem is, given a voxel x and an integer vector v, to compute the first voxel $I(x,v)$ encountered by following a discrete line from x in the direction of v. First, which discrete line should we choose, since there are several ? Here, the good arithmetical properties of the chosen line are not really important. The main problem is to perform a fast computation of a 6−connected discrete line which is close to the continuous half line from x directed by v.

To do so, we introduce a current voxel M initialized to x. In fact, we must initialize M to a voxel close to x on the considered discrete line in such a way that M is not, as is x, a surface voxel, i.e. the leave $L(M)$ of the octee corresponding to M is a complement leave. The choice of an initial M is a bit tricky.

Then we iterate the following procedure until the leaf $L(M)$ is an object leave. We assume for instance that the coordinates of the vector v are all positive.

1. Find the limits $x_{max}, y_{max}, z_{max}$ of the cube corresponding to the leaf $L(M)$ of the octree ;
2. Determine, using integers, which limit will be crossed first from x by following the line in the direction v ;
3. Assume that, say, the limit x_{max} is crossed first. Compute a new voxel M having $x_{max} + 1$ as first coordinate on the discrete line. To insure 6−connectedness, we choose M such that $M.y \leq y_{max}$ and $M.z \leq z_{max}$;
4. Find the new leaf $L(M)$ in the octree data structure.

Let W be the width of the voxels matrix. Since the time for searching a leaf in the octree is at most $\log W$, the time for going over the discrete line is at most $W \log W$. In fact, the required time is generally much less because, when the complement leave $L(M)$ corresponding to M represents a large cube C, first the depth of $L(M)$ is less than $\log W$, and second, we jump directly to the exit of C without considering intermediate voxels. We see here one advantage of the octree data structure: going over a discrete line is fast.

4.2 The two Main Steps of the Method

The purpose of the method is to compute the numbers $B_i(x)$ for each voxel x of the discretized scene, and for i sufficiently large so that $B_i(x)$ is a correct approximation of the solution of Equation 2. Fortunately, the convergence is exponential with respect to i, and it turns out that, in practice, we have an accurate enough estimation for $i = 5, \ldots, 10$.

As the classical radiosity techniques, our method consists in two main steps :

1. Computation, for all voxels x of the scene and for all voxels v of the discrete sphere Σ_R centered at the point x, of the first voxel $I(x, v)$ encountered by following the discrete 6−connected line issued from x through v. The address of the corresponding leave of the octree data structure may be stored.
2. Computation, using the inductive definition of $B_i(x)$ (Equation 3) and the $I(x, v)$ computed during the first step, of the numbers B_i's.

The First Step Note that the result of the first step depends only on the geometry of the scene, and not on the material properties of the objects (numbers $\rho(x)$) nor on the illumination conditions (light sources, emitance). This first step is analogous to the comptation of the *form factors* in classical radiosity methods, and doesn't need to be performed again in case of a change in the illumination conditions. Note the important difference of our method with respect to other radiosity methods, that the first step consists in *integer only* computations.

The complexity of the method is the time L for going over the longest discrete line segment included in the complement of the scene, multiplied by the number $|\Sigma_R|$ of voxels of the considered discrete sphere (which is $O(R^2)$), multiplied by the number N of voxels of the scene.

The Second Step Now we come to the second step which, after the first step, is quite straightforward. The second step is called the *propagation step*. Let us mention that all calculations concerning discrete spheres, including computation of the solid angles $A(f)$, can be performed once for all, independently from the voxel x. Note that we stored the normal vector at each voxel (which is used for computing $\cos\theta(x,v)$) while discretizing the scene, and that this normal vector is computed by continuous techniques (such as the Phong method). An interesting point is that we can, instead of duplicating the variables B_i in order to store B_{i-1} and B_i as Equation 3 suggests, we can use the previously computed $B_i(y)$ instead of $B_{i-1}(y)$ in order to compute $B_i(x)$. The obtained method can also be proved to converge, and, practically speaking, it converges even faster. The complexity of this step is the number i of iterations (typically 5 or 6), multiplied by $|\Sigma_R|$, multiplied by the number of voxels of the scene. Therefore, as we can also observe from experiments, the complexity of the second step is lower than the complexity of the first step. Hence the overall complexity of the method is $LN|\Sigma_R| + iN|\Sigma_R|$.

4.3 About the Space Complexity

The memory cost of the method, as described above, mainly corresponds to the cost of storage of the adresses of $I(x,v)$ for all x and v, which is $N|\Sigma_R|$. This can be managed by storing the octree data structure (hence voxel information) in the RAM, while storing the addresses $I(x,v)$'s on a disk. Indeed, by implementing carefully the algorithm, we can write once for all each address $I(x,v)$ (maybe write blocks of a few MB) during the first step, and then, during the second step, read these addresses as many times as the number i of propagations, always in the same order as the addresses were written. Thus, the disk write/read cost is low, and the memory cost affordable.

5 Experimental Results

First let us describe the computer with which the experiments were made: the processor is 1.2 GHz, we needed 500Mo of RAM, and a 50Go IDE disk. We present 2 experiments with the same 3D scene:

- The first experiment uses a 315*315*190 matrix of voxels, the surfaces being approximates by about 1 million voxels; the radius of the used discrete sphere is 30, so that the cardinality of the discrete sphere (which is the number of directions in space taken into account) is

9194. The amount of memory used is less than 120Mo in RAM to store the octree data structure, about 18Go were written on the disk. The runtime is about 25 hours.
- The second experiment (Figure 1(a)) uses a 420*420*250 matrix of voxels; the radius of the used discrete sphere is 38, so that the cardinality of the discrete sphere is around 15000. The amount of RAM to encode the octree data structure is about 200Mo, and less than 50Go were written on the disk. The runtime is about 72 hours.

Conclusion

We have presented a completely new simulation technique for lighting in a 3D scene made of ideal diffuse reflectors. This method is based on a space voxelization, integer only arithmetic, and give promizing results. However, this is the first paper on this method and a huge amount of work remains to be done, including, first the evaluation of the method as a simulation technique and comparison with classical radiosity methods; second the generalization to an adaptative voxel space by working with higher resolution arround highly curved objects (see [BM02]); third the generalization to specular reflectors and transparency; and fourth arithmetical optimization.

References

[ANF97] E. Andres, Ph. Nehlig, J. Francon, *Tunnel-Free Supercover 3D Polygons and Polyhedra*, Eurographics '97, Budapest, Computer Graphics Forum, ed. Blackwell Publishers, vol. 16, 3, pp. C3-C13, 1997.

[BM00] J. Burguet, R. Malgouyres, *Strong Thinning and Polyhedric Approximation of the surface of a Voxel Object*, Proceedings Discrete Applied Mathematics (DAM) 125(1), pp 93-114, 2002.

[BM02] J. Burguet, R. Malgouyres, *Multiscale Discrete Surfaces*, Proceedings of DGCI'2002, Lecture Notes in Computer Science 2301, Springer, pp 338-349, 2002.

[FVD96] J.D. Foley, A. Van Dam, S.K. Feiner and J.F. Hughes, *Computer Graphics: introduction and practice (second edition in C)*, Addison-Wesley.

[S85] J. Sandor: *Octree Data Structures and Perspective Imagery*, C&G Vol. 9, No. 4, pp.393-405, 1985

[SP94] F.X. Sillon and C. Puech, *Radiosity & Golbal Illumination*, Morgan Kaufmann Publishers, San Francisco, California, 1994.

[SC95] N. Stolte and R. Caubet. *Discrete Ray-Tracing of Huge Voxel Spaces*, Eurographics 95, pages 383-394, Maastricht, August 1995. Blackwell.

[W95] K. Y. Whang et al, *Octree-R: an adaptive octree for efficient ray tracing*, IEEE TVCG, Vol. 1, No. 4, pp. 343-349, 1995

[YCK92] R. Yagel, D. Cohen, and A. Kaufman, *Discrete Ray Tracing*, IEEE Computer Graphics and Applications, September 1992, 19-28.

(a) 25 hours, viewpoint 1.

(b) 72 hours, viewpoint 2.

Fig. 1. Color plates: the living-room scene.