Anisotropic Participating Media at Near Interactive Rates

Juan-Roberto Jiménez[†,‡] and Xavier Pueyo[‡]

[†] GGGJ/Dpto. Informática. Univ. of Jaen
Postal address: Avda. Madrid 35, E-23071 Jaén, Spain.
Phone : +34 953002476
Fax : +34 953002420
E-mail: rjimenez@ujaen.es
[‡] GGG/IIiA. Univ. of Girona
Postal address: Campus de Montilivi, E-17003 Girona, Spain.
E-mail: xavier@ima.udg.es

Abstract

This article presents an algorithm for the realistic simulation, at near interactive rates, of globally illuminated scenes including participating media, isotropic and anisotropic. The proposed algorithm is divided in three phases: particle tracing, reconstruction and rendering. The particle tracing phase is based on the Monte Carlo light tracing. The goal of this phase is to deliver the light energy from the sources, over the scene's object. In the reconstruction phase the algorithm derives one texture for each object in the scene, 2D textures for surfaces and 3D textures for participating media. Each texel represents the outgoing radiance, in a given direction, from a region of the object. The method uses density estimation techniques to compute the textures. In the final phase, the algorithm renders the textures using hardware graphics capabilities.

Categories and Subject Descriptors (according to ACM CCS): G.3 [Probability and Statistics]: Probabilistic algorithms (including Monte Carlo) I.3.7 [Computer Graphics]: Color, shading, shadowing, and texture

1. Introduction

Realistic image synthesis of scenes with participating media requires the simulation of the complex physical and optical aspects of the media. Normally, the non-participating media environments, count clear air or vacuum. Then, for these environments, mainly, the reflection and refraction phenomena must be simulated. To include participating media, in the represented environments, also involves the simulation of the light traversing the medium. This phenomenon is complex due, chiefly, to the additional dimension to represent it, and, due to the continuous change of the propagation direction of the light into the media.

Participating media is a generic term to describe physical phenomena as fog, dusty, clouds, smokes, humidity, etc. In 1994, Rushmeier [15] gives an extended explanation of the participating media application areas. The first presented [3] algorithm simulated media with important simplifications to do the problem computationally treatable, bearing only in mind local illumination. The globally illuminated scenes including participating media may be classified as deterministic or stochastic [14]. The zonal method, is a deterministic method, that constitutes an extension of the classic radiosity algorithm, based on the form factors computation [16]. This algorithm solves the multiple scattering problem taking into account isotropic media. Subsequently, several proposals have been presented to handle anisotropic media. Languenou et al. [9] divide the space of directions in discrete ordinates, Bhate [1] uses spherical harmonics, Stam [19] applies the diffusion equation, or Pérez et al. [13] extend the hierarchical radiosity [17] to include anisotropic media. These solutions involve the use of complex representations and enough memory. On the other hand, there are methods based on the Monte Carlo probabilistic method. These approaches consume less memory, but they used a lot of computation time to eliminate the characteristic noise [8, 12]. Jensen et al. [6] presented a bidirectional ray tracing algorithm based on photon maps. This technique uses density estimation techniques to overcome the noise problem, but the rendering phase is time consuming. Recently, in 2002, Biri et al. [2] have presented a

real time algorithm for fog, but they only use local illumination.

Therefore, the realistic simulation of participating media taking into account global illumination, is a complex problem. And, normally, the time to render an image is high. This rendering time is not interactive but for isotropic media. The proposed global illumination algorithm achieves, nearly, interactive rates, even for anisotropic media, without compromising the quality of the final image.

The article has been structured in the following sections: in all three following sections 2, 3, 4, the phases of the proposed algorithm has been explained. Our preliminary results are commented in section 5 and the conclusions and future work in section 6.

2. Particle tracing phase

The goal of this phase is to distribute the energy from the light sources to the rest of the objects in the scene. Each light source emits a set of particles. These particles are emitted based on the directional distribution function of the light sources. When a particle hits an object in the scene, part of its energy is absorbed, and other part is reflected and/or transmitted. At the end of this phase each object stores a set of particles, one for each hit. The algorithm records the following information for every stored particle: the incoming direction, the emitter of the particle, the interaction point and the incoming energy flux.

In the scene a particle may interact with a surface or a participating medium. If the interaction is with a surface, the reflected ray is chosen between all the outgoing directions on accounting of the BRDF (Bi-directional Reflection Distribution Function). If the particle interacts with a participating medium, first, we must compute the exact interaction point. The geometry of the medium only give us the entry and the exit point of a ray. Therefore, it must be taken into account the optical properties of the medium to know the distance traversed across it. It is possible for a particle, not to intersect with a medium, although it is in his way.

3. Reconstruction phase

The aim of this phase of the algorithm, is to compute the radiated energy from every point in the scene, to a given viewpoint. Therefore, this phase have to be recomputed whenever the viewpoint changes, except for the diffuse surfaces and for the isotropic media. For these objects the reconstruction is only performed once, and, therefore, this step may be considered as a pre-process pass.

This phase is based on the nearest neighbour density estimation technique. The algorithm computes 2D and 3D textures for surfaces and participating media, respectively. Each texture is not directly obtained from the particles set. As an intermediate step, for each object a histogram is constructed. The histogram is one of the more basic density estimation techniques [18]. The use of the histogram, in our algorithm, is similar to the Myskowski proposal [10]. It is used, as a previous step, to efficiently apply the nearest neighbour density estimation technique.

The histogram is computed, directly, from the particle set. The intersection point, of each particle hit, belongs to a histogram cell. The radiosity B is computed, for each cell of the histogram, based on the set of particles stored in it:

$$B = \sum_{\forall p \in cell} L_{o,p} cos\theta \omega_p \qquad (1)$$

where p is a particle, $L_{o,p}$ is the outgoing radiance of the particle p and ω_p is the solid angle related to the particle p.

Once the histogram is totally filled, every texel of the texture is calculated considering the nearest neighbour density estimation. Like the histogram, the texture is a rectangular region. The difference between the histogram and the texture is the resolution. The resolution of the histogram is greater than the texture resolution. The initial value for the texel is obtained from the corresponding region of the histogram, as it is shown in figure 1. The darker area of the histogram is the region corresponding to the texel. The shadowed area is the region to take into account for the computation of the texel value. The area of the region of the histogram corresponding to the texel, is determined by the accumulated radiosity in that region. A given radiosity threshold is, previously, established, for the calculation of the area of the histogram corresponding to each texel. This area is, progressively, increased until the accumulated radiosity reaches the given threshold. For participating media we apply the same algorithm, but the histogram and the 3D texture are divided into a set of voxels instead of texels.

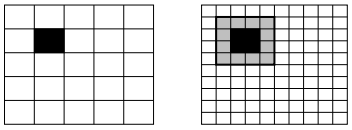


Figure 1: *This figure represents the region of the histogram needed to compute a texel. Left: Texture to compute. Right: Histogram.*

A summed area table (SAT) is used to accelerate the computation of a region of the histogram. The SAT has the same dimensions as the histogram. Each cell of the SAT stores the sum of all the previous values in the histogram:

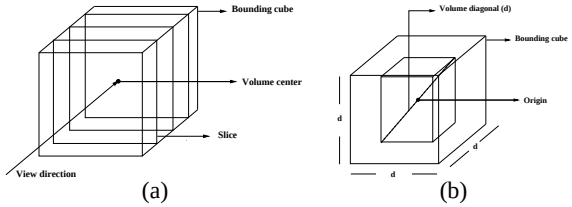
$$Sat(x,y,z) = \sum_{i \le x, j \le y, k \le z} histogram(i,j,k) \qquad (2)$$

Taking into account the SAT, we need only two sums and three subtracts, to compute the accumulation value of whatever region of a 2D histogram. For a 3D histogram, we need three sums and four subtract [4, 5].

4. Rendering phase

Every object of the scene is represented by its corresponding texture. The surfaces are rendered first as being opaque and because they hide any participating medium behind them. We use the OpenGL z-buffer to implement this phase. The 3D hardware textures are used for the rendering of participating media. These textures are divided in slices and are rendered in a back-front order. Each slice must be perpendicular to the view direction (see figure 2(a)). The distance between the different slices determines the transmittance factor of the media $\tau(x_0,x) = exp^{-\int_{x_0}^{x} k_t(u)du}$, where k_t is the extinction coefficient. This transmittance factor is applied using the openGL alpha blending.

To render the slices perpendicularly to the view direction, the algorithm is based on the technique proposed by Gelder et al. [20] for volume visualization; but adapted to the participating media case. Each participating medium is wrapped in a cube of dimensions d^3, being d the diagonal of the medium. The front face of this cube is always perpendicular to the view direction. Consequently, the slices defined over this big cube, are, also, perpendicular to the view direction, as it is shown in figure 2. The medium inside the cube, is properly rotated, using the GL_TEXTURE mode.



(a) (b)

Figure 2: *These figures represent a piece of a participating medium. (a) represents how the participating medium is divided into a set of slices and their direction, and (b) shows the participating medium wrapped by a bounding cube.*

5. Results

The algorithm has been implemented in a Pentium III with Intel processor at 1000 MHz, with a 256Mb memory RAM and a graphical card Nvidia GForce4 Ti 4600 64MB. The Realistic Image Synthesis platform, developed by Martin et al. [11], called SIR(*Sintesis de Imágenes Realistas*) has been used. The scenes have been built using the MGFE [7] format (*Material Geometry Format Extended*). In turn, for the rendering phase we use the graphical library OpenGL, and the GL_TEXTURE_3D extension for the management of 3D textures.

The obtained results achieve near interactive rates. Logically, the rates depend on the geometric complexity of the scene, on the number of glossy surfaces and the anisotropic media. The diffuse surfaces and the isotropic media may be computed in the preprocess pass. Then, they need only one pass for the reconstruction phase, and for successive view points, only the rendering phase time must be considered.

Figure 3 shows two images obtained using our algorithm. The number of emitted particles, in both cases, is one million. Table 1 shows the obtained times for rendering the images in figure 3. The number of slices used, are similar for both images. Whenever the view direction changes, the scene including anisotropic media is rendered in one second, approximately. For isotropic media, the algorithm achieves interactive rates.

	Isotropic	Anisotropic
time	P.: 43.27 s R.: 1.14 s Rd.: 0.08 s	P.: 44.32 s R.: 1.11 s Rd.: 0.07 s

Table 1: *Obtained times for images in figure 3. P., R., and Rd. refer to particle tracing phase, reconstruction phase and rendering phase times, respectively.*

(a) (b)

Figure 3: *Images obtained by our algorithm. The scene includes (a) isotropic, or (b) anisotropic media.*

6. Conclusions and future work

We have presented, in this article, an algorithm for rendering scenes including isotropic and anisotropic participating media, at near interactive rates. The goal of this algorithm has been to achieve efficient results without compromising the quality of the final image. This issue has been attained combining hardware graphics capabilities and density estimation techniques.

One of the aspects to improve, in the future, is the rendering of specular objects. Due to the fact that all the particles are emitted in the preprocess pass, it may occur that there is not enough particles for a given direction, specially, for specular objects. To overcome this problem it may be necessary to use a similar technique to the bidirectional path tracing presented by Lafortune et al. [8], and to shot more particles from the observer, or from the light sources taking into account the current view.

Another aspect we want to improve is the efficiency of the rendering of anisotropic participating media. When the view point changes, the histogram and the *summed area table* must be totally re-calculated, and, certainly, the texture. To obtain the new histogram, we may take advantage from the previous computed histogram, because, normally, the view direction suffers small changes at a time. Our study is, currently, focused on doing part of the reconstruction phase with the hardware graphics card.

Acknowledgements
Part of this work has been financed by projects: TIC2001-226-C02-02 and TIC2001-2099-C03-03 of CICYT (Spain) and DURSI 2001SGR0296 of Generalitat of Catalunya.

References

1. N. Bhate and A. Tokuta. Photorealistic Volume Rendering of Media with Directional Scattering. In *Third Eurographics Workshop on Rendering*, pages 227–245, Bristol, UK, May 1992. 1
2. V. Biri, S. Michelin, and D. Arquès. Real-time animation of realistic fog. In *Rendering Techniques 2002 (Proceedings of the Thirteenth Eurographics Workshop on Rendering)*. ACM Press, 2002. Poster paper. 1
3. J. F. Blinn. Light reflection functions for simulation of clouds and dusty surfaces. In *Computer Graphics (SIGGRAPH '82 Proceedings)*, volume 16, pages 21–29, July 1982. 1
4. F. C. Crow. Summed-area tables for texture mapping. In H. Christiansen, editor, *SIGGRAPH '84 Conference Proceedings (Minneapolis, MN, July 23-27, 1984)*, pages 207–212. ACM, July 1984. 2
5. A. S. Glassner. Multidimensional sum tables. In *Graphics Gems*, pages 376–381. 2
6. H. W. Jensen and P. H. Christensen. Efficient simulation of light transport in scenes with participating media using photon maps. In *Computer Graphics (ACM SIGGRAPH '98 Proceedings)*, pages 311–320, 1998. 1
7. J.R. Jiménez, I. Martín, and F. Pérez. Mgfe: Materials and geometry format extended. http://ima.udg.es/iiia/GGG/doc/mgfehtml/mgfe.shtml. 3
8. E. P. Lafortune and Y. D. Willems. Rendering Participating Media with Bidirectional Path Tracing. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 91–100, New York, NY, 1996. Springer-Verlag/Wien. 1, 3
9. E. Languenou, K. Bouatouch, and M. Chelle. Global Illumination in Presence of Participating Media with General Properties. In *Fifth Eurographics Workshop on Rendering*, pages 69–85, Darmstadt, Germany, June 1994. 1
10. K. Myszkowski. Lighting reconstruction using fast and adaptive density estimation techniques. In Julie Dorsey and Philipp Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 251–262, New York, NY, 1997. Springer Wien. ISBN 3-211-83001-4. 2
11. I. Martín, F. Pérez, and X. Pueyo. The SIR rendering architecture. *Computers & Graphics*, 22(5):601–609, 1998. 3
12. M. Pauly, T. Kollig, and A. Keller. Metropolis light transport for participating media. In B. Peroche and H. Rushmeier, editors, *Rendering Techniques 2000 (Proceedings of the Eleventh Eurographics Workshop on Rendering)*, pages 11–22, New York, NY, 2000. Springer Wien. 1
13. F. Pérez, I. Martín, F. X. Sillion, and X. Pueyo. Acceleration of monte carlo path tracing in general environments. In *Proceedings of Pacific Graphics 2000*, Hong Kong, PRC, October 2000. 1
14. F. Pérez, X. Pueyo, and F. X. Sillion. Global illumination techniques for the simulation of participating media. In Julie Dorsey and Philipp Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 309–320, New York, NY, 1997. Springer Wien. 1
15. H. Rushmeier. Rendering Participating Media: Problems and Solutions from Application Areas. In *Fifth Eurographics Workshop on Rendering*, pages 35–56, Darmstadt, Germany, June 1994. 1
16. H. E. Rushmeier and K. E. Torrance. The Zonal Method for Calculating Light Intensities in the Presence of a Participating Medium. In *Computer Graphics (ACM SIGGRAPH '87 Proceedings)*, volume 21, pages 293–302, July 1987. 1
17. F. X. Sillion. A Unified Hierarchical Algorithm for Global Illumination with Scattering Volumes and Object Clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3), September 1995. 1
18. B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986. 2
19. J. Stam. Multiple Scattering as a Diffusion Process. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 41–50, New York, NY, 1995. Springer-Verlag. 1
20. A. Van Gelder and K. Kim. Direct volume rendering with shading via three-dimensional textures. In *1996 Volume Visualization Symposium*, pages 23–30. IEEE, October 1996. 3