Massively Parallel Radiosity in the Presence of Multiple Isotropic Volume Scattering

Michael S. Langer Pierre Breton Steven W. Zucker
Center for Intelligent Machines McGill University
3480 University St. Montreal, H3A2A7, Canada

Abstract

We present a massively parallel algorithm for computing radiosity in a scene containing an isotropic scattering medium of high albedo. The algorithm is based on a new formalism: a coordinate evolution on the set of light rays in the scene. The evolution reparameterizes rays from their points of origin to their points of termination. Local coordinate systems on the set of light rays are distributed over a two-dimensional lattice of parallel processors, and transformations between coordinate systems are computed using only local communication between processors. The algorithm is implemented on a MasPar-1 which is a SIMD computer having over 2000 processors.

1 Introduction

Massively parallel computers have entered computer graphics research, raising the issue of how to embed the radiosity problem into massively parallel SIMD computing architectures. We propose a new approach to this, based on dynamically solving a coordinate evolution on the set of light rays in a scene, rather than precomputing a set of surface-to-surface visibility functions, or form factors. The result is a massively parallel radiosity algorithm that embeds naturally into a two-dimensional SIMD computing architecture. The algorithm generalizes to scenes containing volume scattering of high albedo, with little extra programming effort.

2 Previous Work

In the absence of a participating medium, surface interreflections have been traditionally modelled by expressing the radiance of each surface point as a weighted sum of the radiances of all the other surface points[19, 10]. The main bottleneck in computing the weights, or *form factors*, is to determine which surface facets are visible from which. This visibility problem may be solved by computing a perspective view of the scene for each surface facet[9]. This computation typically makes use of specialized z-buffer hardware, and the *sequential* access to this hardware remains a fundamental bottleneck in radiosity computations.

When volume scattering fog or clouds are present, the radiosity equation requires significant revision. One approach is to consider surface-volume and volume-volume form factors along with the already present surface-surface form factors, but this requires an enormous increase in the number of form factors and is impractical[18]. A second approach is to consider single scattering only[3, 12, 15, 17], for example, using a two-step ray-tracing approach. First, light that diverges from a point source enters the scattering media and is attenuated exponentially along each ray as a function of optical depth. Second, a viewing position is chosen, and the radiance that arrives at the viewer along a given ray is computed by integrating the source radiance that was scattered from points along that ray. This approach was used to render, for example, automobile headlights in fog and beams of sunlight through a cluttered scene.

A single scattering model is sufficient for low albedo volume scattering. When the albedo is high, however, a single scattering model is inappropriate. For example, milk is a suspension of particles that scatter blue light more than red, yet milk is always white, and this whiteness is due to multiple scattering[4]. Indeed, for many engineering problems in radiative transfer, multiple scattering must be considered to achieve accurate solutions. Present algorithms for computing multiple scattering are serial. (See [11] for a comprehensive review.) In this paper, we present an algorithm which embeds naturally into a SIMD computing architecture.

3 Overview

We first present a new radiosity algorithm for the case in which no participating medium is present, and then generalize the algorithm to include scattering. Consider a surface point $\mathbf{x}_s$ and let $\mathcal{H}(\mathbf{x}_s)$ denote the hemisphere of directions pointing out of the surface at $\mathbf{x}_s$. Let $R(\mathbf{x}, \mathbf{L})$ denote the radiance ($\mathrm{Wm}^{-2}\mathrm{sr}^{-1}$) at position $\mathbf{x}$ and in direction $\mathbf{L}$. For a Lambertian surface, the radiance leaving $\mathbf{x}_s$ does not depend on direction, and hence may be written

$$R(\mathbf{x}_s) = R_{emit}(\mathbf{x}_s) + \frac{\rho(\mathbf{x})}{\pi} \int_{\mathcal{H}(\mathbf{x}_s)} R(\mathbf{x}_s, -\mathbf{L})\ \mathbf{N}(\mathbf{x}_s) \cdot \mathbf{L}\ d\Omega\ .$$

This equation may be solved numerically for each $\mathbf{x}_s$ by using a Jacobi iteration,

$$R^{k+1}(\mathbf{x}_s) = R_{emit}(\mathbf{x}_s) + \frac{\rho(\mathbf{x}_s)}{\pi} \sum_{\mathbf{L}\in\mathcal{H}(\mathbf{x}_s)} R^k(\mathbf{x}_s, -\mathbf{L})\, \mathbf{N}(\mathbf{x}_s)\cdot\mathbf{L}\, \Delta\Omega\ . \quad (1)$$

We solve (1) by compute a *reparameterization* on the set of light rays, from the point of origin of each ray to the point of termination. The theory behind this reparameterization is presented in Section 4. Data structures for representing the rays are specified in Section 5, and the algorithm itself is presented in Section 6. There are two key advantages of the new algorithm. First, it may be embedded into a massively parallel SIMD architecture. This embedding is presented in Section 7. Second, it may be extended to the case in which high albedo volume scattering is present. This case is presented in Section 8.

4 Light Ray Manifold, $\mathcal{M}$

Let $\mathcal{M}$ denote the set of all light rays in a given scene. Local coordinate systems on $\mathcal{M}$ may be defined in a variety of ways. First, consider a local patch $\mathbf{x}(u,v)$ of a surface in the scene. For each point $\mathbf{x}$ on the patch, the set of light rays that originate from $\mathbf{x}$ may be parameterized by a hemisphere of unit vectors $\mathcal{H}(\mathbf{x})$. Since both a hemisphere and a surface patch are two dimensional sets, the set of light rays that originate from the surface patch is four dimensional, i.e. $(u,v)\times\mathcal{H}$.

A similar local coordinate system may be defined by parameterizing the rays that terminate at the surface patch. These two coordinate systems on $\mathcal{M}$ are used in traditional radiosity algorithms. Surfaces in a scene are defined by a set of planar facets, and for each facet, a hemicube of incident [7] or exitant [6] rays is given.

Finally, in [1], the set of light rays in a scene was parameterized using a five dimensional space, $\Re^3\times S^2$, where $\Re^3$ specifies a point through which a ray passes and the unit sphere S^2 specifies the direction of the ray. This parameterization is redundant by one dimension, namely, the dimension of points through which each ray passes.

4.1 Local Coordinates on $\mathcal{M}$

In this paper, an alternative local coordinate system for the set of rays, $\mathcal{M}$, is introduced (see Figure 1). Consider a two-dimensional plane within the scene, for example, the plane $z\equiv z_0$. A given point on this *coordinate plane* is either in free space, inside an object, or on the surface of an object. Each light ray that intersects this plane is specified by four coordinates: two determine the point at which the ray intersects the plane, and two determine the direction of the ray. For example, the light ray that passes through a point (x,y,z_0) in direction $(p,q,1)$ may be parameterized by (x,y,p,q).

Notice that the above parameterization does not cover all the rays in the scene. For example, if a ray were strictly contained in either of the open half spaces $\{z>z_0\}$ or $\{z<z_0\}$, then it would not pass through the plane $z\equiv z_0$. In this sense, the above parameterization defines a *local* coordinate system on $\mathcal{M}$. A set of such local coordinate systems is required to cover the entire set of rays, $\mathcal{M}$. (e.g. coordinate planes, $x\equiv x_0$ and $y\equiv y_0$, are needed to capture light rays that are embedded within $z\equiv z_0$.)

In [13], these coordinate plane parameterizations are used to show that $\mathcal{M}$ is a four dimensional manifold (see [8] for the definition of a manifold). We thus refer to $\mathcal{M}$ as the *light ray manifold.*

4.2 Coordinate Evolution

As the coordinate plane $z\equiv z_0$ is swept through space in the z-direction, a *coordinate evolution* on $\mathcal{M}$ is obtained. For example, consider a light ray traverses both $z\equiv z_0$ and $z\equiv z_1$ (see Figure 1). Let (x,y,p,q) and (x',y',p,q) be the parameterizations of this ray through the two planes respectively, so that

$$(x',y') = (x+(z_1-z_0)p, y+(z_1-z_0)q).$$

In particular, observe that the transformation from the z_0 coordinate plane to the z_1 coordinate plane is *linear.* The linearity observation is crucial because it will allow us to compute a coordinate evolution on $\mathcal{M}$ using a massively parallel SIMD computing architecture.

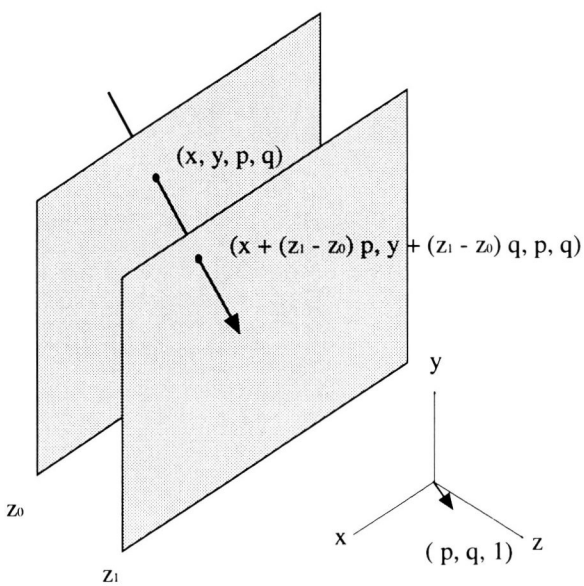

Figure 1: A light ray that is transverse to the coordinate planes, $z\equiv z_0$ and $z\equiv z_1$, and travels in direction $(p,q,1)$. In the plane, $z\equiv z_0$, the ray is parameterized by (x,y,p,q). In the plane $z\equiv z_1$, the ray is parametized by (x',y',p,q), where $(x',y') = (x+(z_1-z_0)p, y+(z_1-z_0)q)$.

5 Data Structures

The view volume of a scene is represented by a $N \times N \times N$ cubic lattice. Nodes in this lattice are of three types: SOLID nodes, SURFACE nodes and FREE nodes. Light rays travel through FREE nodes. Light is absorbed, reflected, and emitted at SURFACE nodes. Light does not reach SOLID nodes.

Scene geometry is defined by a set of SOLID nodes. A non-SOLID node is then a SURFACE node if it is within a given distance M from a SOLID node; otherwise, it is a FREE node. (In our examples, we use $N = 128$ and $M = 5$.) Normals at each SURFACE node correspond to the vector from that node to the center of mass of the FREE nodes in an M-neighborhood of the node.

For each FREE or SURFACE node, $\mathbf{x}$, the finite set of light rays that pass through $\mathbf{x}$ is defined by the nodes on a small cube of diameter $2M$ centered at $\mathbf{x}$. The directions of these rays are defined by the line segments joining $\mathbf{x}$ to points on the six faces of this cube. Quantization errors, or "ray effects", which result from this coarse sampling of the sphere of ray directions are well-known[14, 2, 16].

The light ray cube is analogous to the *hemicube* of [7]; however, there are two important differences. First, the half width of our cube is much smaller ($M = 5$ *vs.* $M = 50$) than that of [7]. Second, a cube is defined at each FREE node, as well as at each SURFACE node; in [7], a hemicube was only defined at each surface facet. By also representing cubes at FREE nodes, the topological structure of the rays $\mathcal{M}$ can be exploited. In particular, a coordinate evolution on $\mathcal{M}$ can be computed on a massively parallel SIMD architecture.

Local coordinates on the sampled light ray manifold are represented as shown in Figure 2. For a given face F of the light ray cube (i.e. there are 6 faces), consider the ith plane in the cubic space lattice that is parallel to F. The local coordinate system $\mathcal{C}_i^F \subset \mathcal{M}$ is the set of light rays that are specified by F and that pass through FREE or SURFACE nodes in plane i,

$$\mathcal{C}_i^F \equiv \{ (u,v,p,q) : (u,v,i) \text{ is a FREE or SURFACE node, and } (p,q,M) \in F \}.$$

Neighboring local coordinate systems may overlap (see Figure 3). For example, consider a FREE node $\mathbf{x} = (x,y,i)$, and a ray passing through this node in direction $\mathbf{L} = (p,q,M)$. This ray could be parameterized in at least three different ways:

$$(u,v,p,q) \in \mathcal{C}_i^F , \quad (u+p, v+q, p, q) \in \mathcal{C}_{i+M}^F ,$$

$$\text{or } (u-p, v-q, p, q) \in \mathcal{C}_{i-M}^F .$$

Note that the transformation from one local coordinate system to the another is linear. In the next section, we present a massively parallel algorithm for transforming from one coordinate system $\mathcal{C}_i^F$ to its neighbor $\mathcal{C}_{i+M}^F$. A sequence of these transformations defines a coordinate evolution.

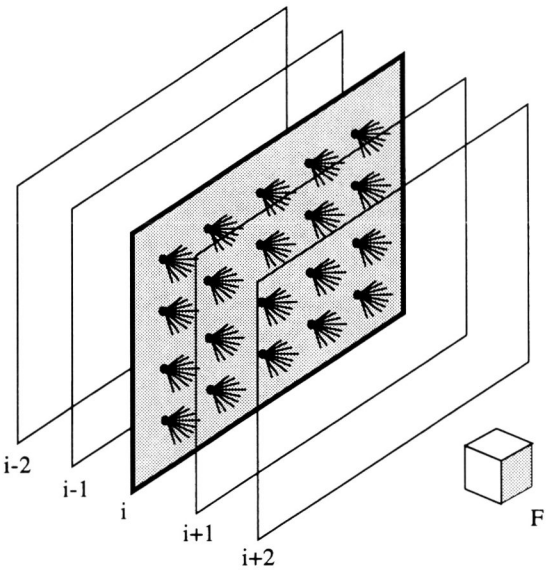

Figure 2: A local coordinate system on the light ray manifold is defined follows. For a given cube face F, consider the ith plane in the space lattice that is parallel to face F. The local coordinate system $\mathcal{C}_i^F$ is the set of light rays that are in the directions specified by F and that pass through FREE or SURFACE nodes in plane i.

6 Algorithm

Consider two neighboring coordinate systems $\mathcal{C}_i^F$ and $\mathcal{C}_{i+M}^F$. Given an estimate, $R^k(\mathbf{x})$, of the surface radiance and an estimate $R_i^k(u,v,p,q)$, of the radiance of rays in $\mathcal{C}_i^F$, an estimate $R_{i+M}^k(u,v,p,q)$ of the radiance of rays in $\mathcal{C}_{i+M}^F$ can be computed by a coordinate transform from $\mathcal{C}_i^F$ to $\mathcal{C}_{i+M}^F$.

```
Local_Transformation( F, i, R_i^k, R_{i+M}^k, R^k ) {
  for all (u,v) in parallel
    x := (u,v,i);
    for all L := (p,q,M)
      case {x - L }
      SOLID : R_{i+M}(u,v,p,q) := 0;
      FREE : R_{i+M}(u,v,p,q) := R_i(u-p, v-q, p, q);
      SURFACE :  if (L · N(x - L) > 0)
                   R_{i+M}(u,v,p,q) := R(x - L);
                 else  R_{i+M}(u,v,p,q) := 0;
}
```

A coordinate evolution is defined by a sequence of coordinate transformations along the three orthogonal axes of the cubic lattice, first in the positive direction and then in the negative direction of each axis. Each of these six sweeps covers the rays defined by a single face of the cube of light ray directions.

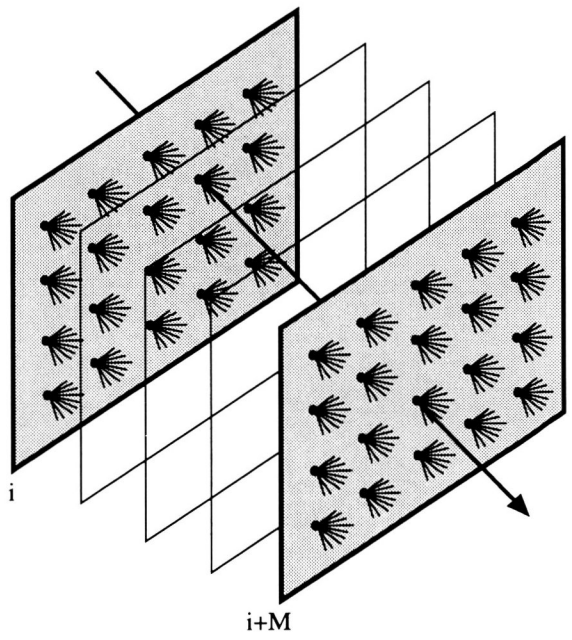

Figure 3: A light ray traverses two neighboring coordinate planes. The tranformation from one local coordinate system to the another is linear.

Coordinate_Evolution(R^k, R^{k+1}) {
  for all cube faces F,
    initialize R_i^k on $\mathcal{C}_i^F$ at $i = 0$;
    for $(i = 0;\ i < N;\ i := i + 1)$ {
      Local_Transformation$(F, i, R_i^k, R_{i+M}^k, R^k)$;
}

Three observations should be made. First, only two coordinate systems, $\mathcal{C}_i^F$ and $\mathcal{C}_{i+M}^F$ need to be represented at a single time. Second, once a ray has reached its point of termination, the surface radiance estimate, R^{k+1}, may be updated using (1). (In our implementation, this update is performed within the procedure **Local_Transformation**.) Third, the coordinate evolution may interpreted as the propagation of radiance along rays, so that the algorithm is just a simulation of the physics of radiosity. The number of iterations of **Coordinate_Evolution** corresponds to the number of surface interreflections.

7 Implementation on MasPar

We implemented the two routines **Local_Transformation** and **Coordinate_Evolution** on a massively parallel computer, a MasPar-1, which is a SIMD machine having 2048 4-bit processing elements or PEs. The PEs form a two dimensional array of size $N_u \times N_v = 64 \times 32$. Each PE is directly connected to its eight neighbors, and the boundary is connected in a toroidal topology. Each PE has 16K of local memory. We rendered scenes of width $N = 128$. An example is shown in Figure 4.

7.1 Memory Costs

The local memory cost to each PE is $O(N^3/N_uN_v + M^2N^2/N_uN_v)$. The two terms are as follows. First, since $N > N_u$ and $N > N_v$, the scene must be wrapped around the PE array N/N_u times in the u direction and N/N_v times in the v direction. This wrapping is facilitated by the toroidal topology of the PE array. Thus, N^2/N_uN_v space columns were represented at each PE and each column was N nodes deep, leading to $O(N^3/N_uN_v)$ space nodes at each PE.

Second, within the procedure **Local_Trans- formation**, $O(M^2)$ rays were required for each space node in $\mathcal{C}_i^F$ and $\mathcal{C}_{i+M}^F$ to represent radiance. Since there were N^2 nodes in each coordinate plane, $O(M^2N^2/N_uN_v)$ rays were represented at each PE.

A number of state variables were represented for each space node. For SURFACE nodes, the variables included the space type, albedo, exitant radiance, surface normal and emittance. For FREE nodes, the state variables included the scattering and absorption coefficients, and for the case of isotropic scattering, the accumulated luminescence - see Section 8).

Recall also that for a given cube face F, the scene must be rotated within the PE array such that each PE contains space columns perpendicular to F. That is, each cube face F defines *intra*-plane coordinates (u, v) as well as an *inter*-plane coordinate i. The rotation is necessary because the PE array is two rather than three dimensional. Fortunately, we were able to devise an inexpensive massively parallel algorithm for performing this rotation. The algorithm requires $O(N)$ memory at each PE, which is insignificant.

Finally, we note that more recent models of the MasPar machine have 16K processors (128×128) and 64K local memory per processor. On such a machine, it would be possible to generate 512×512 images.

7.2 Time Costs

The time cost of the algorithm per iteration is $O(N^3M^2/N_uN_v)$. The inner loop, **Local_Transformation**, is $O(M^2N^2/N_uN_v)$ and the outer loop **Coordinate_Evolution** is $O(N)$. The procedure **Local_Transformation** requires roughly 0.6 seconds when $N = 128$, resulting in a total computation time of roughly 8 minutes per iteration of **Coordinate_Evolution**.

The rotation of the scene had a time complexity of $O(N^3/N_uN_v)$. The operations required for the axis rotation are simple, and in practice the time cost of the rotation was insignificant. When $N = 128$, each rotation took 0.3 seconds.

8 Volume Scattering

Volume scattering is described by an equation relating the radiance arriving at a point in space to the radiance leaving the point[11],

$$\frac{d}{d\epsilon} R(\mathbf{x}+\epsilon\mathbf{L},\mathbf{L}) = (-\zeta(\mathbf{x})-\gamma(\mathbf{x}))\, R(\mathbf{x},\mathbf{L}) + \frac{\zeta(\mathbf{x})}{4\pi} \int_{\mathcal{S}^2} \Phi(\mathbf{L},\mathbf{L}^*)\, R(\mathbf{x},\mathbf{L}^*)\, d\Omega^*, \quad (2)$$

where $\zeta(\mathbf{x})$ is a scattering coefficient, $\gamma(\mathbf{x})$ is an absorption coefficient, and $\Phi(\mathbf{L},\mathbf{L}^*)$ describes scattering anisotropy[3]. Observe that if no volume scattering were present, then $\zeta(\mathbf{x})$ and $\gamma(\mathbf{x})$ are identically zero, and radiance would be conserved along a ray.

One well-known method for solving the scattering equation is the *discrete ordinate method*[5], which was originally developed for a plane slab geometry, that is, a wall of finite thickness but infinite length and width. In this paper, we present a massively parallel algorithm which is similar to the discrete ordinate method, and which applies to general scene geometries. Because of the present memory limitations of our computer, we consider only the isotropic scattering case, that is, surfaces obey Lambert's Law and $\Phi(\mathbf{L},\mathbf{L}^*)$ is constant. The algorithm extends naturally to the case of anisotropic scattering, but with an additional memory cost.

Isotropic scattering may be incorporated into our algorithm as isotropic luminescence. [1] Light energy is temporarily accumulated in a state variable $\Lambda^k(\mathbf{x})$ whenever a light ray passes through $\mathbf{x}$. The energy that is accumulated during iteration k is scattered in iteration $k+1$. The **Local_Transformation** algorithm must be modified slightly. A box is placed around the modified lines of pseudocode.

A key observation is that the time and space complexity of the algorithm are unchanged. This is in sharp contrast to the method proposed in [18], in which the computational costs increased dramatically in the presence of a participating medium.

```
Local_Transformation( F, i, R_i, R_{i+M}, R^k, Λ^k ) {
  for all (u, v) in parallel {
    x := (u, v, i);
    for all L := (p, q, M),
      case {x - L}
      SOLID :   R_{i+M}(u, v, p, q) := 0;
      FREE :
```

```
        R_{i+M}(u,v,p,q) := (ζ(x)/4π) Λ^k(x)
            +(1 − γ(x) − ζ(x)) R_i(u − p, v − q, p, q);
        Λ^{k+1}(x)      += R_i(u − p, v − q, p, q) ΔΩ;
      SURFACE : if (L · N(x − L) > 0)
                  R_{i+M}(u, v, p, q) := R(x − L);
                else R_{i+M}(u, v, p, q) := 0;
  }
```

[1] Non-isotropic scattering is possible in principle, but requires $O(M^2)$ more memory per space node to keep track of the accumulated energy.

An example is shown in Figure 5. Three balls are shown floating in a box that is filled with homogeneous fog. The scene is illuminated by a directed light source that is from above but slightly oblique. The shadows cast by the balls are visible both in space (as tubes) and on the surface. Observe that the deepest ball is the darkest since light is absorbed as it passes through the fog. Five iterations of **Coordinate_Evolution** were used.

9 Conclusion

In this paper, we developed a novel radiosity algorithm which generalizes to the case of multiple isotropic volume scattering. The algorithm is similar in flavour to the discrete ordinates method, but takes advantage of this local geometry of the ray manifold. We defined local coordinate systems on this manifold, embedded these local coordinate systems into a two dimensional lattice of parallel processors, and computed transformations between local coordinate systems. These transformations require only local communication between processors, which is ideal for SIMD architectures e.g. the MasPar-1.

While memory limitations of the MasPar-1 forced us to consider only isotropic scattering, machines having more processors and larger local memory are now entering the marketplace. These machines could be used to generalize our algorithm to the case of non-isotropic scattering. The existence of rendering algorithms that make use of these SIMD machines is a further technological incentive for the development of massively parallel SIMD graphics hardware.

Acknowledgements

This research was supported by grants from NSERC and AFOSR. S.W. Zucker is a Fellow of the Canadian Institute for Advanced Research. The authors would like to thank Nelson Max for his thoughtful comments on an earlier version of this manuscript.

References

[1] J. Arvo and D. Kirk. Fast ray tracing by ray classification. *Computer Graphics*, 21(4):55–64, 1987.

[2] D. Baum, H. Rushmeier, and J. Winget. Improving radiosity solutions through the use of analytically determined form factors. *Computer Graphics*, 23(3):325–334, July 1989.

[3] J. F. Blinn. Light reflection functions for simulation of clouds and dusty surfaces. *Computer Graphics*, 16:21–29, 1982.

[4] C. Bohren. Multiple scattering of light and some of its observable consequence. *American Journal of Physics*, 55(524–533), 1987.

[5] S. Chandresekhar. *Radiative Transfer*. Dover, New York, 1960.

[6] M. F. Cohen, S. E. Chen, J. R. Wallace, and D. P. Greenberg. A progressive refinement approach to fast radiosity image generation. *SIGGRAPH:88*, 22(4):75–84, August 1988.

[7] M. F. Cohen and D. P. Greenberg. The hemicube: A radiosity solution for complex environments. *Computer Graphics*, 19(3):31–40, July 1985.

[8] M. P. DoCarmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Englewood Cliffs, NJ, 1976.

[9] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice.* Addison-Wesley, Reading, Mass., second edition, 1990.

[10] C. M. Goral, K. E. Torrence, D. P. Greenberg, and B. Battaile. Modelling the interaction of light between diffuse surfaces. *SIGGRAPH 84*, 18(3):213–222, July 1984.

[11] J. Howell. Thermal radiation in participating media: The past, the present, and some possible futures. *Journal of Heat Transfer*, 110:1220–1229, November 1988.

[12] J. Kajiya and B. von Hertzen. Ray tracing volume densities. *Computer Graphics*, 18(3):165–174, 1984.

[13] M. S. Langer. *Shading Computations on the Radiation Manifold.* PhD thesis, McGill University, Montréal, October 1994.

[14] K. Lathrop. Ray effects in discrete ordinates equations. *Nuclear Science and Engineering*, 32:357–369, 1968.

[15] N. Max. Light diffusion through clouds and haze. *Computer Vision, Graphics, and Image Processing*, 33:280–292, 1986.

[16] N. Max. Efficient light propagation for multiple anisotropic volume scattering. In *5th Eurographics Workshop on Rendering*, Darmstadt, Germany, June 1994.

[17] T. Nishita, Y. Miyawaki, and E. Nakamae. A shading model for atmospheric scattering considering luminous intensity distribution of light sources. *Computer Graphics*, 21(4):303–310, 1987.

[18] H. E. Rushmere and K. E. Torrance. The zonal method for calculating light intensities in presence of a participating medium. *SIGGRAPH 87*, 21(4):293–302, July 1987.

[19] E. M. Sparrow and R. D. Cess. *Radiation Heat Transfer.* McGraw-Hill, New-York, 1978.

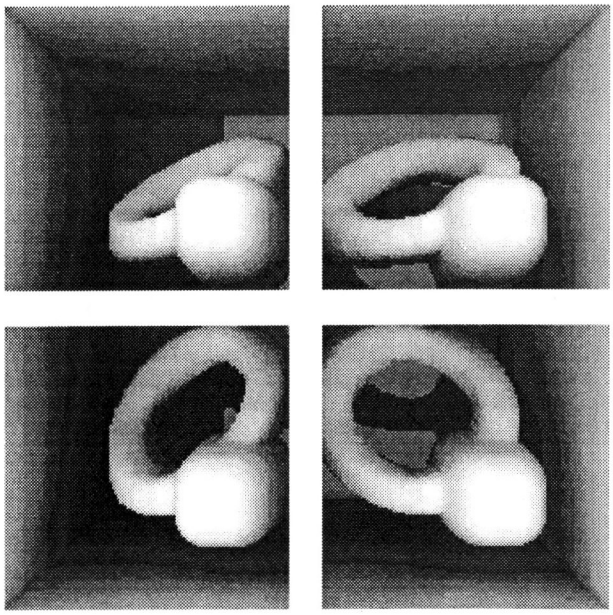

Figure 4: A ring floating in a box is viewed through the top via parallel projection from four different directions. No volume scattering was present. The albedo of all surfaces was 0.9. The light source had two components, ambient and directed, both of which entered the box through the open roof. The ambient component was a uniform hemispheric sky, and the directed component was oblique, from the left. Twelve iterations were used.

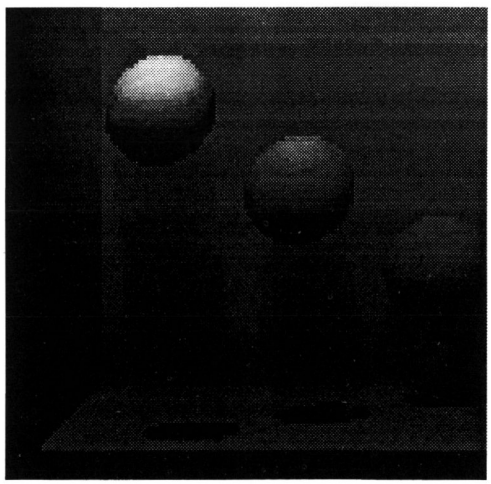

Figure 5: Three balls floating in a box that is filled with fog. The scene is illuminated by a directed light source that is from above but slightly oblique. The shadows cast by the balls are visible both in space (as a tube) and on the surface. Observe that the deepest ball is the darkest since light is absorbed as it passes through the fog.