A Volume Density Optical Model

Peter L. Williams† and Nelson Max‡

†Center for Supercomputing Research & Development, and
National Center for Supercomputing Applications
University of Illinois at Urbana

‡University of California, Davis, and
Lawrence Livermore National Laboratory

Abstract

A simple, but accurate, formal volume density optical model is developed for volume rendering scattered data or scalar fields from the finite element method, as opposed to scanned data sets where material classification is involved. The model is suitable either for ray tracing or projection methods and allows maximum flexibility in setting color and opacity. An expression is derived for the light intensity along a ray in terms of six user-specified transfer functions, three for optical density and three for color. Closed form solutions under several different assumptions are presented including a new exact result for the case that the transfer functions vary piecewise linearly along a ray segment within a cell.

1 Introduction

An exact simulation of light interacting with a volume density or cloud is quite complex and requires the use of Radiative Transport Theory [2, 6]. However, for the purpose of scientific visualization, less complex simulations can be satisfactory.

One of the first computer graphics models for clouds was reported by James Blinn [1] of the Jet Propulsion Lab. He described a method to synthesize an image of the rings of the planet Saturn which consist of clouds of reflective ice particles in orbit about the planet.

Blinn's model deals with the scattering, shadowing and transmission of light propagating through the cloud. It assumes that a ray of light is reflected (scattered) by only a single particle, i.e. multiple reflections are considered negligible. This simplifying assumption will be true if the reflectivity (albedo) of each particle is small. Blinn's model also deals with the shadowing or blocking effect of other particles after a light ray has been scattered by a single particle. And, it deals with the transparency (transmittance) of the cloud layer, that is, the amount of light coming from behind the cloud not blocked off by particles. The brightness or intensity of the cloud at each pixel is calculated by ray integration, that is, by integrating along a ray from the eye passing through the pixel. Many subsequent cloud models are based to some extent on Blinn's model.

Kajiya and Von Herzen [5] give an alternative model which deals with multiple scattering against particles with high albedo; and they further develop Blinn's low albedo model and give a ray tracing algorithm for it. Light propagating through clouds is also discussed by Max [9, 10], Rushmeier and Torrance [12] and Ebert and Parent [3]; however, these techniques are not directly applicable to volume rendering.

Up to this point in the development, the light sources have been outside the cloud and the model has described how the particles in the cloud scatter, absorb and transmit this light.

For the purpose of volume rendering for scientific visualization, a slightly different volume density model is used in which the cloud itself emits light. Two basic models can be used for this. In the first model, the scalar field being visualized is modeled as a cloud of light emitting particles. We will refer to this as the *particle model*. In the other model, the scalar field is represented as a cloud expressed as continuous glowing medium. Each point of the medium both emits and absorbs light. We will refer to this second model as the *continuous model*. The two models are really two different explanations of the same physical phenomenon. For a single transfer

function, which is all the particle model allows, the two models give the same mathematical formula for the derived intensity. By neglecting shadowing on the way in, Blinn's single scattering model turns out to be the same as the particle model if the phase factor is neglected.

The optical model is the most crucial part of a volume renderer but it also can be the most confusing part. Therefore it is important that the underlying model be clearly understood. Earlier models such as in [13, 17] lacked some generality and/or were not easy to comprehend. Our paper presents a new continuous model which is rigorous and quite general, yet is intuitive and easy to understand.

The next section reviews the particle model and also makes some aspects of its derivation more rigorous. Then, in Section 3, we present the continuous optical model for a volume density. This model is suitable either for ray tracing or for projection methods and allows maximum flexibility in setting color and opacity. An expression for the light intensity along a ray through a volume, in terms of six user-specified transfer functions, three for optical density and three for color, is derived. Closed form solutions under several different assumptions are presented, including a new exact result for the case that the transfer functions vary piecewise linearly along a ray segment within a cell. A method is described which allows isosurface shading within a volume rendering.

2 The Particle Model

Paolo Sabella [13] first described a particle model for volume rendering which he called the density emitter model. It is based on Blinn's model but assumes the particles emit their own light, rather than scattering light from a source. Sabella models the density of particles, not the particles themselves. The size of the particle is considered to be small compared to other dimensions so the density of the particles can be regarded as a continuous function.

In Sabella's model, the density of particles at any point $a = (x, y, z)$ is defined by considering a volume element of the cloud dV centered at a. If dV_P is the volume occupied by the particles in dV, then the density function is defined to be $\rho(x, y, z) = dV_P/dV$. If v_p is the volume of a single particle, then the expected number of particles in a region R of the cloud is:

$$N_R = \int_R \frac{dV_P}{v_p} = \int_R \frac{\rho(x, y, z)\, dV}{v_p} \qquad (1)$$

A volume rendered image is created by setting a pixel's color to the intensity of light perceived by the eye along a ray from that pixel to the eye through the volume. Consider a cylinder with cross section σ whose axis is a ray to the eye, parameterized by length t, which enters the cloud at t_1 and exits at t_2. Assume the density function is parameterized along the ray as $\rho(x(t), y(t), z(t))$, or just as $\rho(t)$. An infinitesimal segment S of this cylinder, centered at t and of length dt, has volume $dV = \sigma\, dt$, and contains $N_{dV} = \frac{\rho(t)\, dV}{v_p}$ particles. If the particles are all spheres with radius r, then $v_p = \frac{4}{3}\pi r^3$, and each has projected area πr^2. So if they all glow diffusely on their surfaces with intensity κ, the total light power crossing the front surface of S is

$$\kappa \pi r^2 N_{dV} = \kappa \pi r^2 \rho(t) \sigma\, dt / (\frac{4}{3}\pi r^3) = \frac{3\kappa\sigma}{4r}\rho(t)\, dt$$

The power is distributed over an area σ, so the intensity (power per unit area) contributed by this segment is $\frac{3\kappa}{4r}\rho(t)\, dt$. We have assumed dt is infinitesimal, so that the particles do not occlude each other. But on the path from the interior position t to the front edge t_2 of the cloud, occlusion can take place. To calculate the probability that this ray from t to t_2 is unoccluded, take another cylinder C about it, of radius r, the particle radius. The ray will be unoccluded if there are no particle centers within C. From Equation 1, the expected number N_C of particles inside C is:

$$N_C = \int_C \frac{\rho(x, y, z)\, dV}{v_p} = \int_t^{t_2} \rho(u)\pi r^2\, du / (\frac{4}{3}\pi r^3) = \frac{3}{4r}\int_t^{t_2} \rho(u)\, du$$

If the density is small enough that the chances of mutual overlap are small, the particles can be assumed to be independently distributed. Then the probability $P(0; C)$ that there are no particle centers in the cylinder is given by the Poisson distribution formula $P(0; C) = e^{-N_C} = e^{-\frac{3}{4r}\int_t^{t_2} \rho(u)\, du}$.

Therefore, the intensity of light reaching the eye due to dV is:

$$\frac{3\kappa}{4r}\rho(t)\, dt\, e^{-\frac{3}{4r}\int_t^{t_2} \rho(u)\, du}$$

The total intensity I reaching the eye due to all contributions between t_1 and t_2 is:

$$I = \frac{3\kappa}{4r}\int_{t_1}^{t_2} \rho(t) e^{-\frac{3}{4r}\int_t^{t_2} \rho(u)\, du}\, dt$$

If we assume the intensity at t_1 is zero, that is, there is a black background, and we let $\tau = \frac{3}{4r}$ and $c = \kappa\tau$, we get:

$$I = c\int_{t_1}^{t_2} \rho(t) e^{-\tau\int_t^{t_2} \rho(u)\, du}\, dt \qquad (2)$$

For Equation 2 to hold, the density must be small as required by the Poisson distribution. The density can either be set equal to the scalar field S which is being visualized, or a single user defined transfer function f can

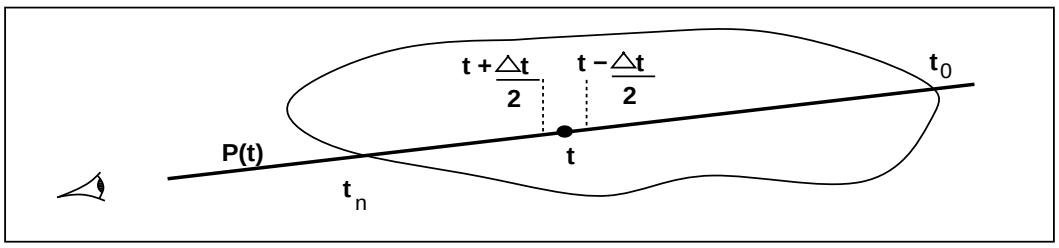

Figure 1: A 2D view of a cloud with a ray P(t) to the eye parameterized by length t.

be used, i.e. $\rho(x, y, z) = f(S(x, y, z))$. Sabella uses numerical methods to estimate the integral in Equation 2 by sampling along the ray.

When the indefinite integral $\int_0^t \rho(u)\,du$ can be tabulated or calculated analytically, and when c and τ are constants, Max, Hanrahan and Crawfis [8] show how Equation 2 can be simplified to a closed form expression which can be evaluated for each cell through which the ray passes. In addition, they take c and τ as vectors with three components, red, green and blue. This has the effect of scaling the single transfer function differently for each of the three components of color. The images created by this method are very accurate renderings of the volume density; however, the process is computationally intensive and really is intended to be implemented in microcode.

It may be possible to modify Sabella's model to include three separate transfer functions for red, green and blue emitted light by assuming that a fraction of the particles emit light of a certain color. Then the density of the red-emitting particles, for example, will vary with a density function $\rho_r(x, y, z)$, and similarly for the blue and green particles. However, these three color particle densities need to be related to the attenuation particle density ρ which appears in the exponent in Equation 2. It seems easier to make this generalization in the continuous model which is described below.

3 The Continuous Model

We now consider the second model, the continuous model for a volume density. This formulation and development is new and has not been presented before. We benefit greatly from the earlier work of Shirley and Tuchman [14] and Wilhelms and Van Gelder [17].

The goal is to provide a simple, but accurate, formal model on which to base direct volume rendering of scalar fields defined on irregular meshes and to maximize the flexibility of use of transfer functions. It is intended for use with scalar field data from the finite element method, or scattered data, as opposed to scanned data sets where material classification is involved. The model is suitable either for ray tracing or for projection methods.

The model is simplified to the bare minimum needed to clearly display the internal structure of the scalar field. No attempt is made to produce a highly realistic simulation of an actual cloud.

3.1 Model Development

In this model, the volume density can be thought of as a luminous or glowing gas cloud, such as neon or a glowing plasma, that selectively absorbs light of certain wavelengths and emits self-generated light. The gas cloud has two physical properties, optical density and chromaticity, both of which are functions of the scalar field being visualized.

The optical density of the gas at any point is wavelength λ dependent and is given by the function $\rho(x, y, z, \lambda) \geq 0$. The chromaticity is specified by a chromaticity function $\kappa(x, y, z, \lambda) > 0$. These two functions are defined in terms of six user-specified transfer functions ρ_r, ρ_g, ρ_b, κ_r, κ_g, κ_b, so for example, $\rho(x, y, z, red) = \rho_r(S(x, y, z))$, where S is the scalar field being visualized.

Let $P(t)$ be a ray to the eye parameterized by length t which enters the cloud at $P(t_0)$ and exits at $P(t_n)$, and let t be a point on the ray centered in the interval $(t + \frac{\Delta t}{2}, t - \frac{\Delta t}{2})$; see Figure 1. Let the notation $\rho(t, \lambda)$ stand for $\rho(P(t), \lambda)$, and similarly for $\kappa(t, \lambda)$.

The meaning of the optical density $\rho(t, \lambda)$ is that, in the limit as Δt goes to zero, $\rho(t, \lambda)\Delta t$ is the fraction of light of wavelength λ entering Δt that is occluded over the distance Δt. The chromaticity $\kappa(t, \lambda)$ has the meaning that, in the limit as Δt approaches zero, $\kappa(t, \lambda)\rho(t, \lambda)\Delta t$ is the intensity of light of wavelength (color) λ emitted at the point $P(t)$. Henceforth, $I(t, \lambda)$ will represent the cumulative intensity of light of wavelength λ at t due to all contributions up to the point t.

The intensity of light reaching $t + \frac{\Delta t}{2}$ is:

$$I(t + \frac{\Delta t}{2}, \lambda) = I(t - \frac{\Delta t}{2}, \lambda)(1 - \rho(t, \lambda)\Delta t) + \kappa(t, \lambda)\rho(t, \lambda)\Delta t$$

Simplifying, we get:

$$\frac{I(t + \frac{\Delta t}{2}, \lambda) - I(t - \frac{\Delta t}{2}, \lambda)}{\Delta t} = -\rho(t, \lambda)I(t - \frac{\Delta t}{2}, \lambda) + \kappa(t, \lambda)\rho(t, \lambda)$$

In the limit as Δt goes to zero, we get:

$$\frac{dI(t, \lambda)}{dt} = -\rho(t, \lambda)I(t, \lambda) + \kappa(t, \lambda)\rho(t, \lambda) \tag{3}$$

Equation 3 is instantiated once for each of the three component wavelengths of light: red, green and blue. In each of these equations, ρ, κ and I are functions only of t; therefore, Equation 3 is really a set of three linear first order differential equations. For example, for red light:

$$\frac{dI_r(t)}{dt} + \rho_r(t)I_r(t) = \kappa_r(t)\rho_r(t)$$

These equations can be solved by numerical methods, for example by the fourth-order Runge-Kutta method, or by linking to a ODE solver subroutine.

Alternatively, by use of the integrating factor $e^{\int_{t_0}^{t} \rho(u,\lambda)\, du}$, we can rewrite Equation 3 as:

$$e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} \frac{dI(t, \lambda)}{dt} + \rho(t, \lambda)e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} I(t, \lambda) = e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} \kappa(t, \lambda)\rho(t, \lambda)$$

then,

$$\frac{d}{dt}\left[e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} I(t, \lambda)\right] = e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} \kappa(t, \lambda)\rho(t, \lambda)$$

Integrating both sides from t_0 to t_n, using the boundary condition that the intensity at t_0 is $I(t_0, \lambda)$, yields:

$$\left[e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} I(t, \lambda)\right]_{t_0}^{t_n} = \int_{t_0}^{t_n} e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} \kappa(t, \lambda)\rho(t, \lambda)\, dt$$

and so,

$$e^{\int_{t_0}^{t_n} \rho(u,\lambda)\, du} I(t_n, \lambda) - e^{\int_{t_0}^{t_0} \rho(u,\lambda)\, du} I(t_0, \lambda) = \int_{t_0}^{t_n} e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} \kappa(t, \lambda)\rho(t, \lambda)\, dt$$

which simplifies to:

$$I(t_n, \lambda) = e^{-\int_{t_0}^{t_n} \rho(t,\lambda)\, dt} \int_{t_0}^{t_n} e^{\int_{t_0}^{t} \rho(u,\lambda)\, du} \kappa(t, \lambda)\rho(t, \lambda)\, dt + I(t_0, \lambda)e^{-\int_{t_0}^{t_n} \rho(t,\lambda)\, dt}$$

(The limits of integration of the integrating factor were chosen so as to satisfy the boundary condition.) By combining the two exponentials in the first term, we get:

$$I(t_n, \lambda) = \int_{t_0}^{t_n} e^{-\int_{t}^{t_n} \rho(u,\lambda)\, du} \kappa(t, \lambda)\rho(t, \lambda)\, dt + I(t_0, \lambda)e^{-\int_{t_0}^{t_n} \rho(t,\lambda)\, dt} \tag{4}$$

Equation 4 can not be solved in closed form for the general case. However, if the transfer functions vary piecewise linearly along a ray segment within a cell, then this equation can be integrated exactly on a cell by cell basis. This solution is given in Section 3.3 after parameter functions are introduced in Section 3.2.

If the chromaticity is assumed to be constant along a ray then, by the same method that Max [8] used for Equation 2 or by a simple transformation of Equation 4, a closed form solution for Equation 4 can be obtained:

$$I(t_n, \lambda) = \kappa(\lambda)(1 - e^{-\int_{t_0}^{t_n} \rho(u,\lambda),\, du}) + I(t_0, \lambda)e^{-\int_{t_0}^{t_n} \rho(t,\lambda)\, dt} \tag{5}$$

If we further assume that both the optical density and chromaticity are constant along a ray, then from Equation 5 or by integrating Equation 3 by separation of variables, we get:

$$I(t_n, \lambda) = \kappa(\lambda)\underbrace{(1 - e^{-\rho(\lambda)(t_n - t_0)})}_{A} + I(t_0, \lambda)\underbrace{e^{-\rho(\lambda)(t_n - t_0)}}_{B} \tag{6}$$

The term labeled B is the transmittance of the region, the fraction of light of wavelength λ entering the region at t_0 that reaches t_n. The opacity α is one minus the transmittance or $1 - e^{-\rho(\lambda)(t_n - t_0)}$ and represents the fraction of light of wavelength λ that enters the region at t_0 that is occluded while passing through the region. Term A represents the attenuation of the light emitted within the region itself. Equation 6 is the alpha compositing formula described by Porter and Duff [11] which they call the *atop* operator.

The restriction that the optical density and/or chromaticity be constant along a ray is not too serious since the cloud is discretized into cells. The ray integration can be evaluated by discretizing the ray in exactly the same way that is used in finite element analysis and then integrating on a cell by cell basis, thus the density and/or chromaticity can vary from cell to cell.

Equation 6 can be evaluated for each cell by letting $\kappa(\lambda)$ and $\rho(\lambda)$ be the average chromaticity and density respectively along the ray between t_1 and t_2, the points where the ray enters and exits the cell:

$$\kappa_{avg}(\lambda) = \frac{\kappa(\lambda, t_1) + \kappa(\lambda, t_2)}{2}$$

and similarly for the optical density. Since it is common to linearly interpolate the scalar field within a cell, this approximation is acceptable. Volume renderers based on Equation 6 are discussed in [14, 19]; both renderers use the visibility ordering algorithm described in [20] . For further efficiency, the opacity α of the cell along the ray,

$$\alpha = 1 - e^{-\rho(\lambda)(t_2 - t_1)} \tag{7}$$

can be approximated by

$$\alpha = \rho(\lambda)(t_2 - t_1) \tag{8}$$

provided $\rho(\lambda)(t_2 - t_1) \ll 1.0$. Photographs of images that compare the use of Equations 7 and 8 are shown in [18]. The calculation can be further simplified if the optical density is assumed to be independent of wavelength, so only four user-defined transfer functions are required. If Equation 7 is used, then the exponential need be evaluated only once per cell.

If it is desired to specify the opacity and color in the range $(0, 1)$, the optical density and chromaticity whose range is $(0, \infty)$ can be normalized to the range $(0, 1)$, as $\hat{\rho} = 1 - e^{-\rho}$ and $\hat{\kappa} = 1 - e^{-\kappa}$, where $\hat{\rho}$ is the normalized density and $\hat{\kappa}$ is the normalized chromaticity.

3.2 Shading, Gradients & Parameter Functions

For shading contour (level) surfaces, the intensity at a point on the surface can be made to vary as a function of the angle between the surface normal vector and a vector to a point light source. One way to incorporate surface shading into the model is discussed at the end of Section 3.3.

The surface normal at any point p on a level surface of a scalar field S is the direction of the gradient of S at p. For tetrahedra, the gradient will be constant throughout the cell. For other types of cells used in the finite element method, the gradient can be computed from the parameter function for the type of cell involved. The parameter function S_c, sometimes referred to in visualization as the interpolation function, gives the value of the scalar field within any given cell. The parameter function may be linear, quadratic, cubic, etc. For example, for a 4 node tetrahedron, the parameter function is linear; and so the scalar field within a cell is given by the parameter function:

$$S_c(x, y, z) = c_1 + c_2 x + c_3 y + c_4 z \tag{9}$$

An appendix gives parameter functions for other common cells and also discusses how the parameter functions relate to the shape or basis functions.

Since the scalar field value is known at the four vertices of the cell and the coordinates of the vertices are known, Equation 9 becomes a set of 4 simultaneous equations which can be solved for c_1, c_2, c_3 and c_4.

The parameter functions will generally not be C^1-continuous at the boundary between cells; and so the gradient will not be C^0-continuous between cells. If the change in the gradient between cells is large then the shading will show anomalies from cell to cell. This can provide useful feedback to the scientist regarding the quality of his/her mesh. However, if smooth shading is desired, an average gradient at each vertex may be calculated by averaging the gradients at the centroids of all cells that share the vertex.[1] The surface normal at any point in a cell can then be calculated by interpolating the gradients at the cell's vertices. A more accurate average vertex gradient can be calculated by weighting the gradients at the surrounding centroids by the inverse of the distance to the centroid. Other methods are described in [4].

3.3 Exact Solution for Linear Parameter and Transfer Functions

Equation 4 can be integrated exactly on a cell by cell basis if the transfer functions vary piecewise linearly along a ray segment within a cell. This can be done as follows.

If the scalar field data has been generated by the finite element method, then, within a cell, the scalar field is given by the parameter function $S_c(x, y, z)$. This is shown for a linear tetrahedron in Equation 9.

We would like to express S_c as a function of t, the ray parameter. In parametric form, a ray through the cell is expressed as:

$$x = \alpha_1 + \alpha_2 t \tag{10}$$

$$y = \beta_1 + \beta_2 t \tag{11}$$

$$z = \gamma_1 + \gamma_2 t \tag{12}$$

where $(\alpha_1, \beta_1, \gamma_1)$ is the point where the ray enters the cell and $(\alpha_2, \beta_2, \gamma_2)$ is a unit vector along the direction of the ray.

Substituting Equations 10, 11 and 12 into Equation 9 gives, for the case of linear tetrahedral elements:

$$S_c(t) = v + wt \tag{13}$$

where $v = c_1 + c_2\alpha_1 + c_3\beta_1 + c_4\gamma_1$ and $w = c_2\alpha_2 + c_3\beta_2 + c_4\gamma_2$.

Now the transfer functions must be considered. Let $s_0 = S_c(t_0)$ be the scalar field value at the point where the ray enters the cell, and $s_n = S_c(t_n)$ be the value at the exit point. If the transfer functions are piecewise linear, as they are in Figure 2, then they can be considered to be composed of $m+1$ piecewise linear intervals (s_0, s_1), (s_1, s_2) , ... , (s_i, s_{i+1}), ... , (s_m, s_n). See Figure 2 where five intervals are shown. The intensity of light at t_n can

[1] When a mesh is rectilinear, a finite difference scheme [15, 7, 16] can be used to approximate the gradient.

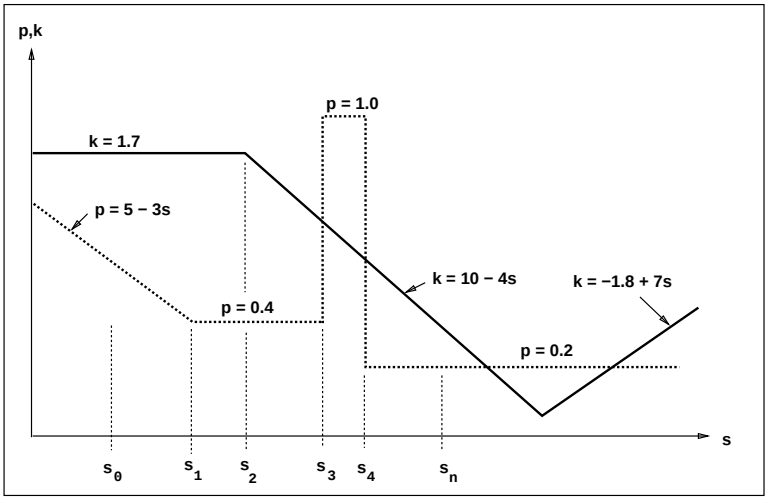

Figure 2: Example piecewise linear transfer functions ρ_r and κ_r, for red light. s_0 and s_n are the values of the scalar field at the points where the ray enters and exits the cell.

be calculated by integrating Equation 4 over each of the $(m+1)$ intervals. The value of t at $s_1, s_2, \ldots, s_m$ can be found from Equation 13. For example: $t_1 = (s_1 - v)/w$.

For red light, the terms involving ρ_r and κ_r in Equation 4 are $\rho_r(s) = a + bs$ and $\kappa_r(s)\rho_r(s) = f + gs + hs^2$ or in terms of t, using Equation 13:

$$\rho_r(t) = a + bv + bwt$$

$$\kappa_r(t)\rho_r(t) = f + gv + hv^2 + gwt + 2hvwt + hw^2t^2$$

over any interval on which κ and ρ are both linear. Hence a table lookup procedure may be used to return the coefficients a, b, f, g and h for any given interval; v and w are calculated from the ray entry and exit points and the parameter function for the cell. For example, for the first interval (s_0, s_1) shown in Figure 2, $a = 5$, $b = -3$, $f = 8.5$, $g = -5.1$ and $h = 0$.

For red light, Equation 4 can then be written as:

$$I_r(t_{i+1}) = \int_{t_i}^{t_{i+1}} e^{-\int_t^{t_{i+1}} (a_i + b_i v + b_i w u)\, du} (f_i + g_i v + h_i v^2 + g_i w t + 2h_i v w t + h_i w^2 t^2)\, dt + I_r(t_i) e^{-\int_{t_i}^{t_{i+1}} (a_i + b_i v + b_i w t)\, dt}$$

simplifying, we get:

$$I_r(t_{i+1}) = \int_{t_i}^{t_{i+1}} e^{-\int_t^{t_{i+1}} (q_1 + q_2 u)\, du} (q_3 + q_4 t + q_5 t^2)\, dt + I_r(t_i) e^{-\int_{t_i}^{t_{i+1}} (q_1 + q_2 t)\, dt}$$

where $q_1 = a_i + b_i v$, $q_2 = b_i w$, $q_3 = f_i + g_i v + h_i v^2$, $q_4 = g_i w + 2h_i v w$ and $q_5 = h_i w^2$.

The second term can be integrated easily to give:

$$I_r(t_i) e^{-(q_1 t_{i+1} + \frac{q_2 t_{i+1}^2}{2} - q_1 t_i - \frac{q_2 t_i^2}{2})}$$

The first term can be evaluated by completing the square of the exponent and then using integration by parts. We verified our hand calculations with *Mathematica*, version 2.0, and report here the *Mathematica* output for the first term:

$$\frac{q_2 q_4 - q_1 q_5 + q_2 q_5 t_{i+1}}{q_2^2} - \frac{q_2 q_4 - q_1 q_5 + q_2 q_5 t_i}{q_2^2} e^{(t_i - t_{i+1})(2q_1 + q_2 t_i + q_2 t_{i+1})/2} + \frac{\sqrt{\frac{\pi}{2}}(q_2^2 q_3 - q_1 q_2 q_4 + q_1^2 q_5 - q_2 q_5)}{q_2^{2.5} e^{(q_1 + q_2 t_{i+1})^2/(2q_2)}} \times \left(\mathrm{Erfi}\left(\frac{q_1 + q_2 t_{i+1}}{\sqrt{2q_2}}\right) - \mathrm{Erfi}\left(\frac{q_1 + q_2 t_i}{\sqrt{2q_2}}\right)\right)$$

In this expression, the argument to Erfi is either real, when $q_2 > 0$, undefined if $q_2 = 0$, or pure imaginary, when $q_2 < 0$. Erfi(x) is shorthand for two functions, depending on whether x is real or pure imaginary. If x is real, $\mathrm{Erfi}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{u^2}\, du$, while if x is pure imaginary, of the form $x = ib$, with real b, $\mathrm{Erfi}(ib) = i\frac{2}{\sqrt{\pi}} \int_0^b e^{-u^2}\, du$. When $q_2 < 0$, the i in the latter expression cancels the i in the denominator $q_2^{2.5}$. Thus, we need only prepare one dimensional tables for these two integrals, or use subroutines to approximate them. (The second integral is closely related to the integral of the Gaussian error function, for which subroutines and tables already exist.)

If $q_2 = 0$, then the first term takes a different form, and integration by parts or *Mathematica* gives:

$$\frac{(q_1^2 q_3 - q_1 q_4 + 2q_5 + q_1^2 q_4 t_{i+1} - 2q_1 q_5 t_{i+1} + q_1^2 q_5 t_{i+1}^2)}{q_1^3} - \frac{(q_1^2 q_3 - q_1 q_4 + 2q_5 + q_1^2 q_4 t_i - 2q_1 q_5 t_i + q_1^2 q_5 t_i^2) e^{q_1(t_i - t_{i+1})}}{q_1^3}$$

This is a new result. It remains to be seen if these

expressions can be evaluated so as to permit interactive rendering. Certainly, they can be used to generate benchmark images against which images generated by approximate solutions, such as those given in Equations 5 and 6, can be compared. And, they can be used for the generation of presentation quality images. Accurate approximations need to be investigated.

Surface shading is helpful for understanding the orientation of (iso)surfaces that may appear in a volume rendered image. However, for the remainder of the image shading is not appropriate and may even make the image harder to understand. An interval in the density map which produces an isosurface is distinctive, for example a tall narrow rectangular pulse or a delta function. If such intervals are flagged, surface shading can be turned on only for those intervals.

4 Conclusion

We have presented a continuum-based volume density optical model specifically for direct volume rendering of scattered data or scalar fields from the finite element method, as opposed to scanned data sets where material classification is involved. The model has been developed so as to be intuitive and also to maximize the flexibility of use of transfer functions. Six user-specified transfer functions, three for optical density and three for color, are permitted. The model is suitable either for ray tracing or for projection methods.

Wilhelms and Van Gelder [17] outline a continuous model and develop differential equations for cumulative intensity and transmittance based on it. Our model benefits from their development, but also differs from their model in a number of respects. For example, in our model the opacity does not appear as a factor in the denominator of the cumulative intensity; and color intensity is linked to opacity, i.e. color and optical density are multiplicative factors. The value of the former property is that the cumulative intensity is well behaved for very small or zero opacity. The latter property means that if the transfer functions are defined such that (a) the optical density at a point in the volume is very small, i.e. the medium is almost totally transparent at that point, and (b) the color intensity is maximum at that point, then the overall contribution will be minimal. These differences need further investigation.

Closed form solutions under several different assumptions have been presented, including a new result for the case when the transfer functions vary piecewise linearly along a ray segment within a cell. This new exact solution needs to be implemented and the rendering time and image quality compared with those resulting from implementations of Equations 5 and 6 and with results from other volume density models. Also, the utility of the three density transfer functions introduced by the continuous model needs to be investigated.

Acknowledgements

Helpful conversations were had with Roger Crawfis, Michael Heath, Paul Saylor and Peter Shirley. John Gray's and Tom DeBoni's assistance with Mathematica is appreciated. The first author is grateful to Peter Shirley for encouraging him to undertake this project. This work was partially supported by the U. S. Department of Energy under Grant DE-FG02-85-ER25001 at the Center for Supercomputing Research and Development, and contract number W-7405-ENG-48 at Lawrence Livermore National Laboratory, the Air Force Office of Scientific Research Grants AFOSR-90-0044, and Sun Microsystems Inc.

Appendix

The parameter function for a 4 node tetrahedron is given in Section 3.2. For the other three most common 3D elements (cells), the parameter functions are as follows. For an undistorted 10 node (quadratic) tetrahedron:

$$S_c(x,y,z) = c_1+c_2x+c_3y+c_4z+c_5x^2+c_6xy+c_7y^2+c_8yz+c_9z^2+c_{10}xz$$

For an 8 node (trilinear) brick:

$$S_c(x,y,z) = c_1 + c_2x + c_3y + c_4z + c_5xy + c_6yz + c_7xz + c_8xyz$$

For a 20 node undistorted (quadratic) brick:

$$\begin{aligned} S_c(x,y,z) \quad = \quad & c_1 + c_2x + c_3y + c_4z + c_5xy + c_6xz + c_7yz + \\ & c_8x^2 + c_9y^2 + c_{10}z^2 + c_{11}xyz + c_{12}x^2y + \\ & c_{13}xy^2 + c_{14}x^2z + c_{15}xz^2 + c_{16}y^2z + \\ & c_{17}yz^2 + c_{18}x^2yz + c_{19}xy^2z + c_{20}xyz^2 \end{aligned}$$

When the elements are distorted by the use of an isoparametric mapping function, then the mapping function needs to be inverted and the interpolation performed in the undistorted element using the parameter function. (In order to assure convergence in the finite element method, the mapping function must be invertible.) The topic of visualization of distorted elements is not dealt with here but it is an important issue that needs to be faced.

The parameter function can in general be expressed in terms of the scalar values s_i at the i vertices of the cell:

$$S_c(x,y,z) = \sum_i N_i(x,y,z)s_i$$

For a linear tetrahedron, this becomes:

$$S_c(x,y,z) = N_1(x,y,z)s_1+N_2(x,y,z)s_2+N_3(x,y,z)s_3+N_4(x,y,z)s_4$$

The coefficients $N_i(x,y,z)$ are referred to as the *shape* or *basis functions* for the cell. Thye are polynomials of the form discussed above; for example on the 4 node tetrahedron they are linear.

References

[1] Blinn, Jim. Light Reflection Functions for Simulation of Clouds and Dusty Surfaces. *ACM SIGGRAPH Comput. Gr. 16 3* (July 1982), 21-29.

[2] Chandrasekhar, S. *Radiative Transfer*. Dover, New York, 1960.

[3] Ebert, David and Parent, Richard. Rendering and Animation of Gaseous Phenomena by Combining Fast Volume and Scanline A-buffer Techniques. *ACM SIGGRAPH Comput. Gr. 24 4* (July 1990), 357–363.

[4] Gallagher, Richard and Nagtegaal, Joop. An Efficient 3D Visualization Technique for Finite Element Models and Other Coarse Volumes. *ACM SIGGRAPH Comput. Gr. 23 3* (July 1989), 185–192.

[5] Kajiya, James and Von Herzen, Brian. Ray Tracing Volume Densities. *ACM SIGGRAPH Comput. Gr. 18 4* (July 1984), 165-174.

[6] Krueger, Wolfgang. The Application of Transport Theory to Visualization of 3D Scalar Data Fields. *Proc. IEEE Visualization '90* (Oct. 1990), 273–280.

[7] Levoy, Marc. Display of Surfaces from Volume Data. *IEEE Comput. Graph. Appl. 8 3* (May 1988) 29-37.

[8] Max, Nelson, Hanrahan, Pat, and Crawfis, Roger. Area and Volume Coherence for Efficient Visualization of 3D Scalar Functions. *San Diego Workshop on Volume Visualization, Comput. Gr. 24 5* (Dec 1990), 27–33.

[9] Max, Nelson. Atmospheric Illumination and Shadows. *ACM SIGGRAPH Comput. Gr. 20 4* (Aug. 1986), 117–124.

[10] Max, Nelson. Light Diffusion through Clouds and Haze. *Comput Vision, Graph. and Image Proc. 33* (March 1986), 280–292.

[11] Porter, Thomas and Duff, Tom. Compositing Digital Images. *ACM SIGGRAPH Comput. Gr. 18 3* (July 1984), 253–259.

[12] Rushmeier, Holly and Torrance, Kenneth. The Zonal Method for Calculating Light Intensities in the Presence of a Participating Medium. *ACM SIGGRAPH Comput. Gr. 21 4* (July 1987), 293–302.

[13] Sabella, Paolo. A Rendering Algorithm for Visualizing 3D Scalar Fields. *ACM SIGGRAPH Comput. Gr. 22 4* (July 1988), 51-58.

[14] Shirley, Peter and Tuchman, Allan. A Polygonal Approximation to Direct Scalar Volume Rendering. *San Diego Workshop on Volume Visualization, Comput. Gr. 24 5* (Dec 1990), 63-70.

[15] Upson, Craig and Keeler, Michael. V-BUFFER: Visible Volume Rendering. *ACM SIGGRAPH Comput. Gr. 22 4* (July 1988), 59-64.

[16] Westover, Lee. Interactive Volume Rendering. *Proceedings Chapel Hill Workshop of Volume Visualization* (May 1989), 9-16.

[17] Wilhelms, Jane and Van Gelder, Allen. A Coherent Projection Approach for Direct Volume Rendering. *ACM SIGGRAPH Comput. Gr. 25 4* (July 1991), 275-284.

[18] Williams, Peter. Interactive Direct Volume Rendering of Curvilinear and Unstructured Data. PhD thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1992, C.S.R.D. Report No. 1243.

[19] Williams, Peter. Interactive Splatting of Nonrectilinear Volumes. To appear in *Proc. IEEE Visualization '92* (Oct. 1992).

[20] Williams, Peter. Visibility Ordering Meshed Polyhedra. *ACM Trans. on Graphics 11 2* (April 1992), 103–126.