Fast Visualization of Volume Emissions using Conservative Subdivision

Mats Holmström

Swedish Institute of Space Physics, PO Box 812, SE-98128 Kiruna, Sweden

Abstract

A hierarchical volume rendering algorithm is presented. Volume emissions in optically thin media are considered. A sparse hierarchical wavelet-based representation of the directional volume emission rate function is constructed top-down and then projected onto the image plane. This makes the algorithm fast, since the number of operations is proportional to the number of coefficients in the representation. Also, the sparsity minimizes the algorithm's memory requirement. The representation is constructed using conservative subdivision, making the projection onto a lower dimensional representation trivial. The error is controlled by a threshold parameter, allowing a direct trade-off between speed and accuracy. The algorithm is especially well suited when the directional volume emission rate is computationally expensive to evaluate, since the sparse representation minimizes the number of function evaluations for rendering a volume with a specified accuracy. Since a sparse oct-tree is constructed for a specific view point the method is best suited to situations where one image is to be generated from each view point.

Key words: Volume Rendering, Subdivision, Oct-tree

1 Introduction

Here we consider computer imaging of volume emissions. For visible light we are imaging the flux of photons in the direction of the observer. In computer graphics this process is denoted volume rendering [12], but the problem is not limited to visible light. Other applications, that inspired this work, are X-ray and energetic neutral atom imaging [8,9]. For these two applications, the extraction of parameters from non-linear emission models require the generation of many images (for different parameters) and the directional volume emission

Email address: matsh@irf.se (Mats Holmström).

rate is given by a function that is computationally expensive to evaluate. The directional volume emission rate is the number of photons (or atoms) that are produced per volume per time unit, as a function of position and (emission) direction. The unit can for example be $[1/(\mathrm{m}^3 \text{ sterad s})]$.

We make the assumption that the medium through which the emissions propagate is optically thin, i.e. there is no attenuation of the emissions. Pure volume emissions are considered, in the sense that no surfaces are present. Extensions of the presented algorithms to surfaces and opaque media are possible, but not discussed in this work. Some desirable properties of a volume rendering algorithm are:

- Speed. Minimize the number of evaluations of the directional volume emission rate function. This is especially important if each evaluation is computationally expensive, as is the case if the function is given by a complicated numerical or analytical model.
- Error control. It is of course desirable to minimize the error, but for all rendering algorithms this comes at the expense of computational time. For visualization applications the error requirements are usually not that stringent. It is enough if the error is small enough not to be visually detected. In scientific applications, such as parameter extraction from models, the situation is different. Here the comparison of simulated and observed images require that the error in the volume rendering is smaller than the observational error, so that the rendering does not restrict the achievable accuracy of the extracted parameters. In all applications it is desirable to easily be able to control the error (and indirectly, the computational time).
- Minimal memory requirements. In what follows, we assume that the number of pixels in the final image is $\mathcal{O}(N^2)$, and that a fine grid on the imaged volume has $\mathcal{O}(N^3)$ grid points. An algorithm should not require the storage of the directional volume emission rate on a fine grid, $\mathcal{O}(N^3)$ storage.[1] Ideally the memory requirement should be proportional to the number of pixels in the final image, $\mathcal{O}(N^2)$ storage.

The method we propose uses a wavelet-based representation of the directional volume emission rate, generated by conservative subdivision — a representation with the property that the coefficients can directly be projected on a representation in the image plane (coefficient splatting) in constant time per coefficient (actually with just one arithmetic operation per coefficient, since the projection is a summing of the coefficients due to the conservation property of the representation). This, along with a top-down approach when constructing the representation, makes the computational work and memory requirements

[1] The big-oh notation ($\mathcal{O}$) is intentionally used rather loosely here, since we for simplicity have assumed that the volume has the same extensions in all directions, and that the whole volume is projected onto the view plane.

proportional to N_s, the number of significant coefficients in the representation. Thus, the algorithm is fast, and it is also conceptually simple. The subdivision is denoted conservative since the volume integral of the represented function is conserved when a cell is subdivided, and when it is projected. This is a natural property when representing physical properties such as photon flux or density.

For comparison, Fourier-based rendering [17] reduces the computational complexity of the straight forward rendering algorithm [4] from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2 \log N)$, but this is still potentially more expensive than the $\mathcal{O}(N_s)$ algorithms that are possible using a wavelet approach. This approach also provides error estimates and error control, automatically.

The proposed algorithm has similarities to that presented by Keller [13], in that correlations between pixels are used to increase speed, but Keller's algorithm is one-dimensional (along scan lines) and is solely in the image plane. He also uses stochastic (Monte Carlo) sampling for pixel irradiances. Since the aim of our approach is to control the error, a stochastic method is not appropriate. Another algorithm that is adaptive in the two-dimensional image plane is presented by Bao, Jin and Peng [1] for global illumination.

One can view the proposed method as an extension of the hierarchical oct-tree representation used by Laur and Hanrahan [14]. They introduced a first order representation (mip-map) and assembled the whole tree before the rendering of the basis function foot prints onto the image plane. The contributions of this work, compared to the mip-map, is that

- The full pyramid representation is never needed since we build the sparse oct-tree top-down.
- The coefficients are projected onto the image plane before a two-dimensional inverse transform that produces the image (coefficient splatting, used by Lippert [16]).
- The order of the spatial accuracy of the algorithm is determined by the order of the subdivision algorithm used. The numerical examples that we present uses quadratic (third order) subdivision, but the method easily generalizes to higher order, and we can find a trade-off between the increased spatial accuracy (smaller oct-tree) and the increased cost for the subdivision of each cell (more cells in the subdivision stencil).

The first two properties ensures that the running time is kept proportional to the number of nodes in the sparse oct-tree, N_s.

Several wavelet-based algorithms for volume rendering have been published [19,7,16,5,6,3], but the wavelet bases used have not been suited for the integration needed when projecting the intensities onto the image plane, requiring expensive transformations to point values on a fine grid, projections of basis function

footprints, or numerical quadrature. Lippert [16] presents a wavelet-based algorithm that uses splatting of wavelet coefficients, but the basis is such that the necessary computations for each splatted coefficient makes the projection expensive. Also, the representation is built from a fine grid, so the work is proportional to N^3. We can note that the construction of a wavelet representation from a fine grid (bottom up) is not necessarily a big problem if one is to generate many images (from different view points) using the same representation, e.g., for an animation, or for tomographic reconstruction, since the $\mathcal{O}(N^3)$ work will be amortized over all the generated images. If however, one only generates one image using the representation, the $\mathcal{O}(N^3)$ work will be prohibitively expensive.

2 Theory

First of all, we define the volume rendering problem, and a canonical problem is introduced to simplify the presentation of different algorithms. Then we introduce a hierarchical representation for fast volume rendering. When a variable is used both with and without boldface, standard typeface denotes the vector's magnitude, e.g., the distance to the origin is $r = |\mathbf{r}|$.

Assume that the directional volume emission rate from position $\mathbf{r}$ in the direction of $\mathbf{d}$ (unit length) is given by $g(\mathbf{r}, \mathbf{d})$ [1/(m^3 sterad s)]. These volume emissions can for example be photons or atoms. Then the radiance at the position $\mathbf{r}$ from the direction of $\mathbf{d}$ (unit length) is

$$f(\mathbf{r}, \mathbf{d}) = \int_0^\infty g(\mathbf{r} + s\mathbf{d}, -\mathbf{d})\, ds \qquad [1/(\mathrm{m}^2 \text{ sterad s})], \tag{1}$$

i.e. the radiance is given by an integration of the directional volume emission rate along a line of sight (LOS) in the view direction, $\mathbf{d}$, as illustrated in Figure 1. An introduction to radiometric quantities can be found in [15].

Fig. 1. The radiance, f, at the position $\mathbf{r}$, from the direction $\mathbf{d}$, is computed by integration of the directional volume emission rate, $g(s)$, along a line of sight.

From now on we will assume that the observer is located at the origin of our coordinate system. Then the emission direction that we are interested in is always directed toward the origin, $\mathbf{d} = -\mathbf{r}/r$, and we can simplify the notation by dropping g's directional dependence, $g(s\mathbf{d}) \equiv g(\mathbf{r} + s\mathbf{d}, -\mathbf{d})$, and f's

dependence on position, $f(\mathbf{d}) \equiv f(\mathbf{r}, \mathbf{d})$, so that (1) becomes

$$f(\mathbf{d}) = \int_0^\infty g(s\mathbf{d})\, ds \qquad [1/(\mathrm{m}^2 \text{ sterad s})], \tag{2}$$

Then the irradiance from a finite solid angle Ω, e.g., corresponding to an image pixel, can be formulated in two ways,

(1) as a two-dimensional integral of the radiance over the solid angle,

$$F(\Omega) = \int_\Omega f(\mathbf{d})\, d\Omega \quad [1/(\mathrm{m}^2 \text{ s})], \qquad \text{or} \tag{3}$$

(2) as a three-dimensional integral of the directional volume emission rate over the volume corresponding to the solid angle Ω,

$$F(\Omega) = \int_{\mathcal{V}} \frac{g(\mathbf{r})}{r^2}\, dV \quad [1/(\mathrm{m}^2 \text{ s})], \tag{4}$$

where $\mathcal{V}$ is the conic volume corresponding to Ω and the decrease in solid angle subtended by the receiver, as seen from $\mathbf{r}$, results in the factor $1/r^2$. The geometry is shown in Figure 2. The two formulations are of course mathematically

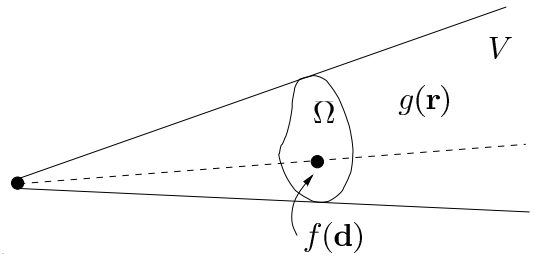

Fig. 2. The irradiance can either be computed by two-dimensional integration of the radiance, f, over all directions, $\mathbf{d}$, contained in the solid angle Ω, or by three-dimensional integration of the directional volume emission rate, g, over the volume corresponding to the solid angle.

equivalent, but they represent two different approaches when constructing algorithms for solving the volume rendering problem. We can think of F as received surface flux on an area element at the origin, perpendicular to the center of the solid angle. We have assumed that Ω is small and neglected a factor $\cos\theta$ in the integration, where θ is the angle between $\mathbf{r}$ and the normal of the receiving area element. For a single image pixel, i, with a corresponding solid angle, Ω_i, centered around $\mathbf{d}_i$, we thus have two approaches to compute the average radiance,

$$f_i = F(\Omega_i)/\Omega_i, \quad [1/(\mathrm{m}^2 \text{ sterad s})].$$

Denote the approach corresponding to (3) by LOS and that corresponding to (4) by VOL.

In practice, regardless of the chosen approach, the integrals are evaluated by numerical quadrature. Using the midpoint quadrature rule for LOS in (3), we have the approximation $f_i \approx f(\mathbf{d}_i)$. This is the straight forward, classical, way of generating an image of a radiation field, where we for each pixel do a line of sight (LOS) integration in the direction corresponding to a pixel, $\mathbf{d}_i$.

The solid angles corresponding to each pixel in an images are defined by the chosen view perspective and discretization of the image plane. Different projections are shown in Figure 3, along with image plane grids that define the pixels. Regardless of projection, the problem can be seen as a projection of a

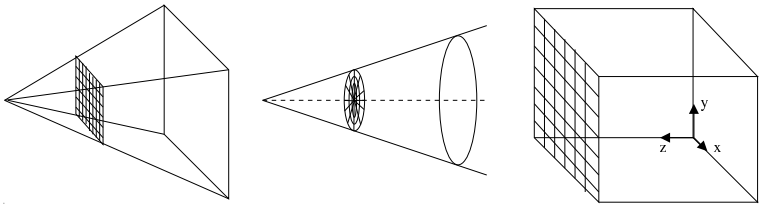

Fig. 3. Examples of different projections of volumes onto image planes. From left to right; perspective, polar and canonical. The image planes, with pixels, are indicated by grids. If we define a closest and farthest distance for the perspective and polar projections, we can coordinate transform both to the canonical projection. In the polar case, the canonical unit cube will be periodic in the azimuth direction.

three-dimensional volume onto a two-dimensional plane along lines of sights. By defining the problem as a canonical projection of the unit cube onto the xy-plane along the z-axis, the algorithms presented are general, regardless of the projection. To adapt an algorithm to a specific projection we can either include an extra coordinate transform (e.g., frustum to cube, or cone to periodic cube) or we can reformulate the algorithm directly for the new geometry.

In the rest of the presentation we use this canonical projection of the unit cube onto the unit square to simplify comparison and presentation of algorithms for volume rendering. Denote the directional volume emission rate in the z direction by $g(x, y, z)$, then the image radiance is

$$f(x, y) = \int_0^1 g(x, y, z)\, dz. \tag{5}$$

The challenge is to compute a representation of f on the grid of image pixels as quickly as possible (for our purposes, quickly means using as few evaluations of g as possible) for a specified error in the approximation of f.

To further simplify the discussion, we consider the analogue of (5) for a two-

dimensional directional volume emission rate. Then the radiance is

$$f(x) = \int_0^1 g(x, z)\, dz, \tag{6}$$

where the image now is a one-dimensional function, $f(x)$. This lower dimensional problem is only introduced for clarity in the presentation of the ideas behind the different volume rendering algorithms. Later on, the numerical results are presented for the full three-dimensional problem (5). The pixels then are intervals on the x-axis, and pixel i's irradiance is

$$f_i = \int_{x_{i-1}}^{x_i} f(x)\, dx = \int_{x_{i-1}}^{x_i} \int_0^1 g(x, z)\, dz\, dx, \tag{7}$$

an integration over the rectangle $[x_{i-1}, x_i] \times [0, 1]$ in the xz-plane, shown in Figure 4.

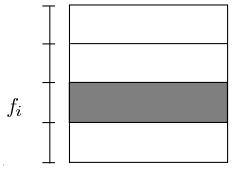

Fig. 4. The area of integration corresponding to a pixel's irradiance for the model problem with a two-dimensional directional volume emission rate. This corresponds to a one-dimensional image defined in (6).

We now consider the two approaches, LOS (3) and VOL (4), for computing the pixel irradiance (7). A straight forward computation with the LOS in the center of the pixel and a uniform step length in the z-direction results in the leftmost tiling of the unit square shown in Figure 5. Each f_i is computed by numerical quadrature along the z-axis. It is easy to improve (in terms of accuracy for a given number of evaluations of the directional volume emission rate) on this basic LOS algorithm. First of all we can use a numerical integration with adaptive step length in the z-direction. Secondly, we can use several LOS for each pixel (super sampling [4]). A tiling using these two improvements is shown in the middle of Figure 5. The drawback of the LOS approach is that we can never achieve an optimal tiling (function representation) for functions that have small regions where there are large changes in the directional volume emission rate. The limitation of using LOS forces us to have several LOS close to each other even in regions where the function is smooth.

By using a wavelet-based hierarchical representation, we can achieve refinement in the representation *only* in the regions where the function has large

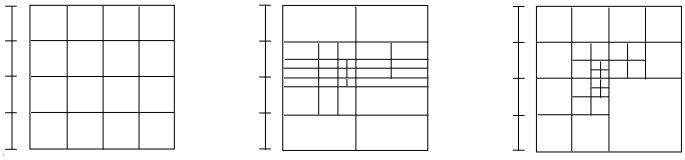

Fig. 5. Different algorithms for solving the model problem with a two-dimensional directional volume emission rate 6 can be illustrated by different tilings of the unit square. From left to right we have; a standard uniform step length LOS integration, an adaptive step length LOS integration with several LOS per pixel, and a hierarchical representation of the function.

variations. A quad-tree example is shown rightmost in Figure 5. The idea is to *first* build the representation, top-down (coarse to fine grid), *then* project it onto a lower dimensional hierarchical representation in the image plane, and *finally* transform to pixel irradiances, f_i. This is the chosen approach in this work, and there are several advantages. The total work is proportional to the number of cells, N_s, in the representation, and the errors of the computed f_i:s are automatically controlled by a threshold parameter, ϵ. The representation will be generated by conservative subdivision, a transform that is described in the next section. We then present the full algorithm in detail.

2.1 A Sparse Hierarchical Function Representation

First we present a short introduction to conservative subdivision, following Donoho [2]. This is sometimes called average-interpolating subdivision but the former name seems more informative. We define cell averages of a one-dimensional function $u(x)$ as

$$u_{j,k} = \frac{1}{\Delta x} \int_{x_{j,k-1/2}}^{x_{j,k+1/2}} u(x)\, dx.$$

Here j denotes scale and k position, i.e. $x_{j,k} = 2^{-j}k$. The cell size is $\Delta x = x_{j,k+1/2} - x_{j,k-1/2}$.

If we, given cell averages on a coarser scale $u_{j,k}$, want to find cell averages on a finer scale $u_{j+1,k}$ (half the cell size) we find the (unique) quadratic polynomial that has integrals $u_{j,k-1}$, $u_{j,k}$ and $u_{j,k+1}$; and then find $u_{j+1,2k}$ and $u_{j+1,2k+1}$ by integrating the quadratic. By repeating this recursively we can achieve a representation on an arbitrarily fine scale. We can also use higher order, even, polynomials. Increasing the order of the polynomial will increase the work, but improve compression of smooth solutions. Boundaries are easily handled by the transform, we simply use the closest available quadratic. The subdivision is very fast (a few arithmetic operation for splitting a cell in two).

If we want to do the opposite, go from a fine $u_{j+1,k}$ to a coarse grid $u_{j,k}$. We have no choice due to conservation,

$$u_{j,k} = \frac{1}{2}\left(u_{j+1,2k} + u_{j+1,2k+1}\right).$$

Now the difference between original cell averages $u_{j+1,k}$ and averages computed by subdivision from the coarser scale $u_{j,k}$ can be viewed as wavelet coefficients (or details)

$$d_{j,k} = \tilde{u}_{j+1,2k} - u_{j+1,2k}$$

where $\tilde{u}_{j+1,2k}$ is the prediction, computed by subdivision.

To construct a sparse hierarchical representation we threshold the wavelet coefficients, i.e. we remove a coefficient if $|d_{j,k}| < \epsilon$. In practice we start at a coarsest scale and stop the transform when $|d_{j,k}| < \epsilon$ for a cell. This then corresponds to a tree representation (a binary tree with the N_s cells as leafs). The extension to higher-dimensional Cartesian grids is straight forward. In two-dimensions, at each level, we subdivide along rows, then along columns and stop if the magnitude of all three wavelet coefficients $< \epsilon$. This gives us a quad-tree, and is illustrated in Figure 6. In three-dimensions we use an oct-tree.

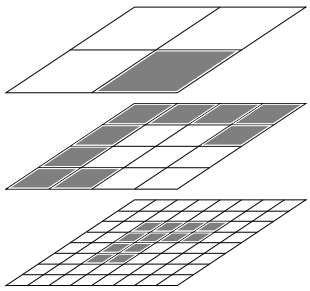

Fig. 6. An illustration of a sparse function representation on a three level hierarchy of two-dimensional grids. The cells corresponding to significant coefficients are shaded. The total number of significant coefficients, N_s, is 22 in this example.

Given the three-dimensional sparse hierarchical function representation, we now reduce the dimension of the representation by one. This is done by a projection along one of the coordinate axes and results in a new, lower-dimensional, sparse hierarchical representation. This operation is particularly simple since our representation consists of cell averages. We simply project each of the levels separately to get the lower dimension representation, as illustrated in Figure 7. Due to the linearity of the transform this projection is exact, in the sense that an inverse transform to the finest grid, followed by a projection, gives exactly the same result, as a projection followed by an

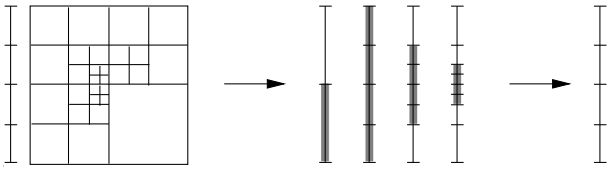

Fig. 7. The projection of a sparse two-dimensional hierarchical representation onto a one-dimensional one. Finally, we subdivide or sum, to get the pixel irradiance.

inverse transform. After this projection, we subdivide or sum the cells in the representation, depending on if the cell size is larger or smaller than the pixel size, to arrive at the pixel irradiances, f_i.

An implementation of the method has some options. First of all, we can choose the quadrature rule used for computing cell averages when constructing the sparse representation. There is a trade-off between the number of function evaluations and the accuracy of the approximation. In all numerical examples we have used the midpoint rule, where the cell average is approximated by the function value at the center of the cell. A disadvantage is that there is no reuse of function values from coarser levels. Using for example the three-dimensional version of Simpson's rule [18], all function evaluations can be reused on a finer level, as shown in Figure 8. Secondly, we can increase the

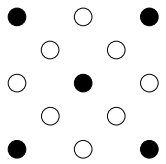

Fig. 8. A two-dimensional illustration that function evaluations can be reused for the multi-dimensional version of Simpson's quadrature rule. Solid circles denote points where the function is evaluated. Circles denote evaluation points at the next finer level.

order of the subdivision rule that we use, i.e. the order of the polynomial that define the subdivision. In our numerical experiments, we have used quadratic subdivision. The order should match the order of the quadrature rule. Also, more general, we can use other methods than conservative subdivision for subdividing a cell.

There are several reasons why conservative subdivision is a transform that is well suited to volume rendering applications. The conservation property is natural, since the directional volume emission rate is a quantity that should be preserved between scales. The conservation property also makes fast and exact projection (splatting) possible. Also, the transform is simple. The main advantages of the hierarchical method for volume rendering are that:

- Total work and memory requirement is proportional to N_s.
- Accuracy is specified by a threshold parameter, ϵ.

- Progressive rendering is possible, i.e. coefficients can be projected ordered by their size. This means that an initial coarse image can be gradually refined.

But if the function is equally smooth everywhere the method will require the same amount of work as a uniform grid method, since the representation will be a uniform grid. The sparser the representation is compared to a uniform grid on the finest level, the more we save (when $N_s \ll N$). A restriction is that the coarsest grid must be fine enough to allow the refinement to capture small features.

These advantages of such a hierarchical representation were also exploited for solving time-dependent partial differential equations in [10], and especially the conservation property in [11].

A drawback of the proposed method is that only projections along the coordinate axes are fast. This is due to the fact that the representation can only be projected directly along these axes. A projection along any other direction would require the computation of basis function footprints. Thus, as mentioned in the Introduction, the method is best suited for the case when one wants to generate one image for a given directional volume emission rate function.

3 Numerical Experiments

To verify and illustrate the performance of the proposed method we solve the canonical problem (5) using the hierarchical method (VOL) and the standard line of sight integration (LOS). The performance measure will be the maximum error in the rendered image (relative to the maximum value), as a function of the number of evaluations of the directional volume emission rate function. As a test function we use

$$f(x,y,z) = \left(\sin(5x)\sin(7y)\sin(11z)/10\right)^2 + e^{-\alpha\left(x-\sqrt{2}/3\right)^2/\sqrt{2}-\alpha\left(y-\sqrt{7}/5\right)^2\sqrt{2}-\alpha\left(z-\sqrt{6}/5\right)^2\sqrt{3}}, \tag{8}$$

on the unit cube, a smooth function with a peak. The width of the Gaussian was chosen as $\alpha = 100$. We note that the width of this peak will influence the comparison between the methods. The sparse VOL method will be favored by a narrow peak (large α) since it only refines the representation close to the peak. On the other hand, if the function is equally smooth everywhere, VOL reduces to the standard LOS method. The value of α used here results in a peak that is reasonably broad, as can be seen in Figure 9, where the result for the hierarchical method is compared to the exact solution for a 64×64 pixel image.

Note that only 4032 function evaluations (less than one per pixel) gives a 6% error. Thus, for generating images that are almost indistinguishable, visually,

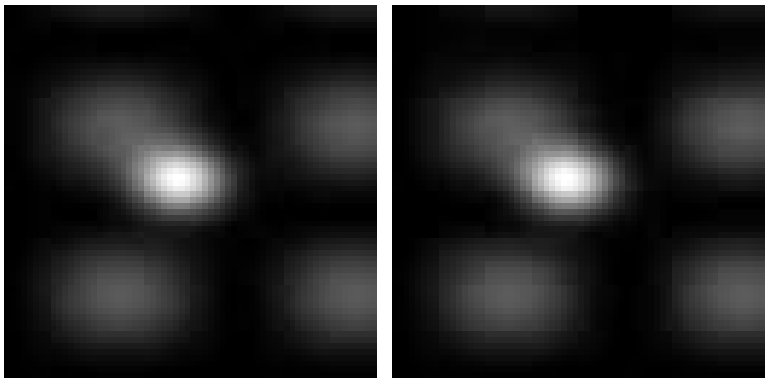

Fig. 9. The reference, 64×64 pixel, image to the left, and a hierarchical image to the right, using 4032 function evaluations (less than one per pixel). The relative maximum error is 6%. The directional volume emission rate function is given in (8).

a very modest number of evaluations of the directional volume emission rate function are needed.

Now, using VOL, we can vary the error by changing the threshold parameter, ϵ, thereby of course changing the number of function evaluations. In Figure 10 the error as a function of the number of function evaluations is compared for LOS and VOL. We see that VOL is about an order of magnitude faster than fix step LOS. The error is 1% using 20 000 function evaluations (5 per pixel). For LOS, we used super-sampling, i.e. 1, 4 and 16 number of lines of sights per

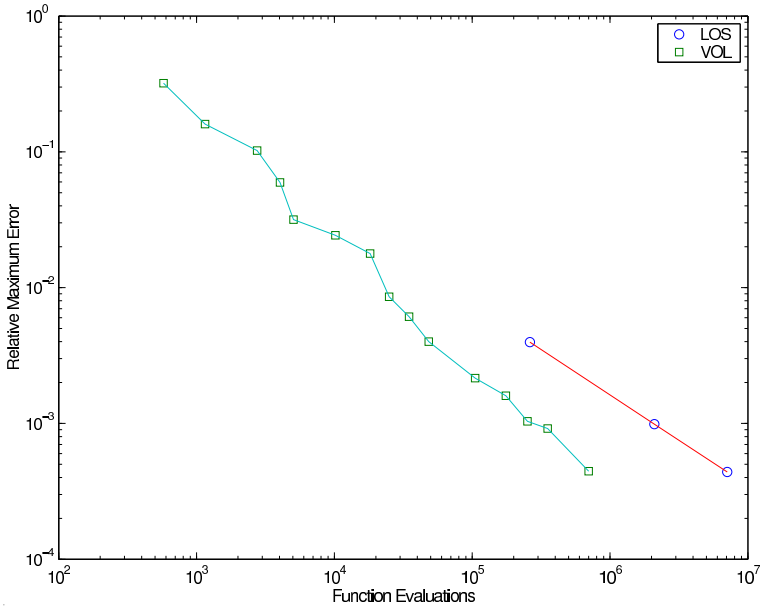

Fig. 10. Relative maximum error for the sparse hierarchical method (VOL) and the line of sight method (LOS) as a function of the number of evaluations of the directional volume emission rate function.

pixel. Note that the error curve is almost linear for VOL in this diagram, over the whole span of errors, from 0.35 to 0.0004. Thus, by varying the threshold, we can achieve any required accuracy. From the slope of the line, the order of VOL seems greater than for LOS, so the gain of using VOL will be larger the higher the wanted accuracy is. That the accuracy is directly controlled by the threshold parameter, ϵ, is illustrated in Figure 11, where we have the error as a function of ϵ.

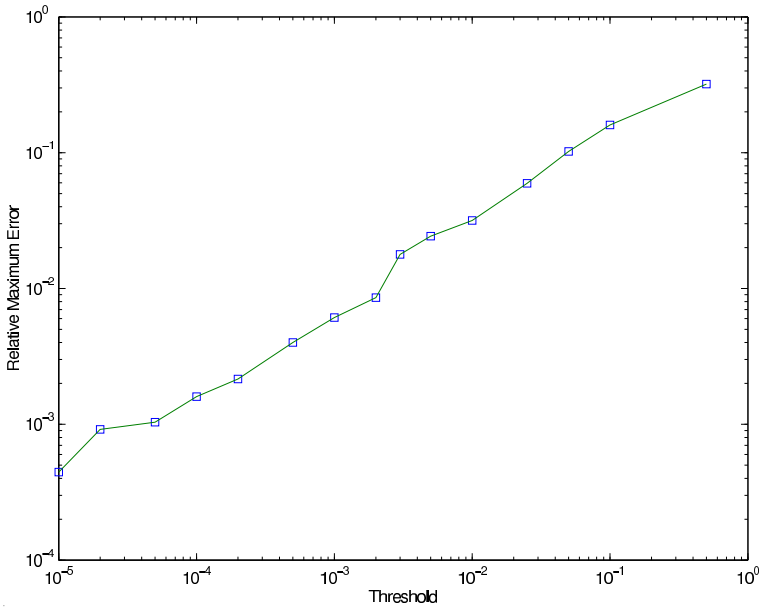

Fig. 11. Relative maximum error for the sparse hierarchical method (VOL) as a function of the specified threshold value, ϵ.

Thus, not only does the sparse representation use almost an order of magnitude less function evaluations to achieve the same error, the error is also decreasing faster when we use more evaluations. We can also note that by adjusting ϵ we can trade error for speed over a large range, as shown in Figure 11.

4 Conclusions

There are many advantages of using a sparse, hierarchical, representation of the directional volume emission rate function for volume rendering. First of all, the number of needed coefficients in the representation can be minimized, since the representation is adapted to the function. Secondly, the error can be controlled by a threshold parameter, enabling a direct trade-off between time and error. Also, the representation enables progressive rendering. Yet another advantage is that the method is directly applicable to higher dimensional problems. For example, in scientific applications there is often the extra dimension

of energy. The method then uses a projection of a four-dimensional sparse representation onto a three-dimensional sparse representation. The advantage of using conservative subdivision for the representation of the directional volume emission rate is that the projection onto the lower dimensional representation (in the image plane) is trivial (and exact) for this specific representation. We emphasize again that the number of operations for the whole rendering problem is proportional to N_s, the number of coefficients in the representation, with a small constant.

Acknowledgements

I thank Timothy Sergienko for helping me understand the nomenclature of radiometry.

References

[1] Hujun Bao, Xiaogang Jin, and Qunsheng Peng. A progressive radiosity algorithm based on piecewise polynomial intensity distribution. *Computers & Graphics*, 21(3):281–288, 1997.

[2] D. L. Donoho. Smooth wavelet decompositions with blocky coefficient kernels. In L. L. Schumaker and G. Webb, editors, *Recent Advances in Wavelet Analysis*, pages 259–308. Academic Press, 1993.

[3] T. Ertl, R. Westermann, and R. Grosso. Multiresolution and hierarchical methods for the visualization of volume data. *Future Generation Computer Systems*, 15(1):31–42, 1999.

[4] Andrew S. Glassner, editor. *An Introduction to Ray Tracing*. Academic Press, 1989.

[5] M. H. Gross, L. Lippert, R. Dittrich, and S. Häring. Wavelet-based volume rendering. *Computers & Graphics*, 21(2):237–252, 1997.

[6] M. H. Gross, L. Lippert, A. Dreger, and R. Koch. A new method to approximate the volume-rendering equation using wavelet bases and piecewise polynomials. *Computers & Graphics*, 19(1):47–62, 1995.

[7] Roberto Grosso, Christoph Lürig, and Thomas Ertl. The multilevel finite element method for adaptive mesh optimization and visualization of volume data. In R. Yagel and H. Hagen, editors, *8th IEEE Visualization Conference*, pages 387–394. IEEE Computer Society Press, 1997.

[8] M. Holmström, S. Barabash, and E. Kallio. X-ray imaging of the solar wind-Mars interaction. *Geophysical Research Letters*, 28(7):1287–1290, 2001.

[9] M. Holmström, S. Barabash, and E. Kallio. Energetic neutral atoms at Mars: I. Imaging of solar wind protons. *Journal of Geophysical Research*, 107(A10):1277, 2002. doi: 10.1029/2001JA000325.

[10] Mats Holmström. Solving hyperbolic PDEs using interpolating wavelets. *SIAM Journal on Scientific Computing*, 21(2):405–420, 1999.

[11] Mats Holmström. Solving hyperbolic PDEs using conservative subdivision schemes. In P. Neittaanmki, T. Tiihonen, and P. Tarvainen, editors, *Proceedings of the 3rd European Conference: Numerical Mathematics and Advanced Applications*, pages 542–547. World Scientific, 2000.

[12] Arie E. Kaufman. Volume visualization. *ACM Computing Surveys*, 28(1):165–167, 1996.

[13] Alexander Keller. Hierarchical Monte Carlo image synthesis. *Mathematics and Computers in Simulation*, 55:79–92, 2001.

[14] David Laur and Pat Hanrahan. Hierarchical splatting: A progressive refinement algorithm for volume rendering. *Computer Graphics*, 25(4):285–288, 1991.

[15] Kuo-Nan Liou. *An Introduction to Atmospheric Radiation*, volume 26 of *International Geophysics Series*, pages 3–6. Academic Press, 1980.

[16] L. Lippert and M. H. Gross. Fast wavelet based volume rendering by accumulation of transparent texture maps. *EUROGRAPHICS '95*, 14(3):431–443, 1995.

[17] Tom Malzbender. Fourier volume rendering. *ACM Transactions on Graphics*, 12(3):233–250, 1993.

[18] A. H. Stroud. *Approximate Calculation of Multiple Integrals*. Prentice Hall, 1971.

[19] Rüdiger Westermann. Compression domain rendering of time-resolved volume data. In G. M. Nielson and D. Silver, editors, *6th IEEE Visualization Conference*, pages 168–175. IEEE Computer Society Press, 1995.