Efficient Data Structures for Volume Rendering of Wavelet-Compressed Data

Roberto Grosso and Thomas Ertl and Joachim Aschoff
Lehrstuhl für Graphische Datenverarbeitung (IMMD 9)
Universität Erlangen-Nürnberg
{grosso,ertl}@informatik.uni-erlangen.de

Abstract

Volume rendering is compute intensive and requires huge amounts of memory. Many methods were proposed to improve performance, but the problems of dealing with large data sets still remain. Muraki was the first to propose the decomposition of the volume density function into a pyramidal representation by means of the 3D discrete wavelet transform with orthonormal bases, and then render the corresponding basis functions weighted by the wavelet coefficients. Depending on the approximation properties of the basis functions used a large number of coefficients can be neglected while still achieving a good representation of the volume data, which results in very high compression rates. However, due to the pyramidal character of the discrete wavelet transform, a very large number of basis functions have to be evaluated in order to approximate the underlying volume density function at any point. This fact increases the rendering time drastically. Furthermore, the access rate to a data value is very slow for typical data structures for compressed data, which combined with an inefficient traversal of the pyramidal structures leads to even larger rendering times. In this paper we propose a data structure for sparse data representation, which does not exploit the full compression potential but combined with an intelligent pyramid traversal allows for a significant acceleration of the rendering time. All other known acceleration techniques to solve the volume integral can be used additionally, such as adaptive integration step, space leaping, early ray termination, etc.

1 Volume Rendering

Volume rendering as a visualization technique is based on the transport equation of light for a medium without scattering. This approximation is known in the volume rendering community as the *emission–absorption* or *density emitter model* [Sab88]. Here, the volume is assumed to be filled with light-emitting particles which are controlled by a density distribution function. This model is not suitable for describing natural phenomena, but it is very useful for generating comprehensive images of three-dimensional scalar fields. The transport equation [Kru91] can be solved by simple ray integration. Denoting the source coefficient by q, the absorption coefficient by κ, and defining s to be the parameter along the ray, the intensity at s is given by

$$I(s) = I(s_0)e^{-\tau(s_0,s)} + \int_{s_0}^{s} q(s')e^{-\tau(s',s)}ds' \, ,$$

with the optical depth $\tau(s_1, s_2) = \int_{s_1}^{s_2} \kappa(s)ds$. $I(s_0)$ is determined by the boundary conditions where the ray enters the volume at position s_0. In order to solve this equation numerically one has to discretize along the ray. Taking samples at points s_k along the ray and introducing the transparency $\theta_k = e^{-\tau(s_{k-1},s_k)}$ between s_{k-1} and s_k and the attenuated source

$b_k = \int_{s_{k-1}}^{s_k} q(s')e^{-\tau(s',s_k)}ds'$ the discrete form of the equation is

$$I(s_n) = I(s_{n-1})\theta_n + b_n = \sum_{k=0}^{n} b_k \prod_{j=k+1}^{n} \theta_j \,,$$

with $b_0 = I(s_0)$. Evaluated from *back–to–front* the accumulated intensity I is obtained:

$$I = \theta_k \times I + b_k$$

Performing the summation from *front–to–back* the accumulated intensity and the accumulated transparency α are given by:

$$I = I + b_k \times \alpha \,, \quad \alpha = \theta_k \times \alpha \,.$$

These are the well known *compositing* relations for intensity and transparency [PD84].

2 Wavelets

The importance of orthonormal basis of wavelets and multiresolution analysis resides in their hierarchical nature, which offers a mathematical framework for describing functions at different levels of resolution. Using basis functions (or coordinate systems) with good approximations properties, i.e. with many vanishing moments, one can represent functions by keeping only the important coordinates (*features*) and discarding all others. In this way, functions can be compressed by representing them in a new coordinate system of fewer dimensions while maintaining their characteristic behavior. This section discusses aspects of wavelets relevant to our application of volume rendering. Details on wavelet theory can be found in [Dau92, Chu92].

2.1 Orthonormal Wavelets and Multiresolution Analysis

A multiresolution analysis can be thought of as a ladder of approximating closed subspaces $(V_j)_{j \epsilon Z}$ of $L^2(\Re)$. The functions in these subspaces have well defined scaling and translation properties. Furthermore, there exists a function $\phi \,\epsilon\, V_0$ such that $\{\phi_{j,n},\, n \epsilon Z\}$, with $\phi_{j,n}(x) = 2^{-j/2}\phi(2^{-j}x - n)$, is an orthonormal basis of V_j, $\forall j \epsilon\, Z$. Under these conditions one can construct an orthonormal wavelet basis $\{\psi_{j,n};\, j, n \,\epsilon\, Z\}$, with $\psi_{j,n}(x) = 2^{-j/2}\psi(2^{-j}x - n)$, such that for any function f in $L^2(\Re)$

$$P_{j-1} f = P_j f + \sum_{n \epsilon Z} < f, \psi_{j,n} > \psi_{j,n} \,, \tag{1}$$

where P_j is the orthogonal projection onto V_j

$$P_j f = \sum_{n} < f, \phi_{j,n} > \phi_{j,n} \,.$$

The function ψ is sometimes called the *mother* wavelet. The projections $P_j f$ onto the subspaces V_j corresponds to the different levels of resolution at which the function f can be described. These projections contain the *smooth* information of f at a given level. The projections

$$Q_j f = \sum_{n \epsilon Z} < f, \psi_{j,n} > \psi_{j,n}$$

onto the subspaces W_j spanned by the $\psi_{j,n}$ represent the *detail* information of f required to move from one resolution approximation subspace to the next finer one. Equation (1) is the wavelet decomposition of the function f. The *scaling* function ϕ satisfies the *two-scale* relation

$$\phi = \sum_{n} h_n \phi_{-1,n} \,, \tag{2}$$

which is a discrete *low-pass filter* operation with the filter $\{h_n\}_{n \epsilon Z}$. For higher dimensions we use bases which are tensor products of the one-dimensional case.

2.2 A Filtering Scheme with Exact Reconstruction

We start with a scale approximation $f^{j-1} = P_{j-1}f$ of a function f in V_{j-1} and decompose it into a coarser approximation in V_j. Due to the fact that $V_{j-1} = V_j \oplus W_j$, we have $f^{j-1} = f^j + \delta^j$, where $\delta^j = Q_j f$. In terms of the orthonormal bases $\{\phi_{j,n}\}_{n\epsilon Z}$ and $\{\psi_{j,n}\}_{n\epsilon Z}$, we have

$$f^{j-1} = \sum_n c_n^{j-1}\phi_{j-1,n}\,, \quad f^j = \sum_n c_n^j \phi_{j,n}\,, \quad \delta^j = \sum_n d_n^j \phi_{j,n}\,, \tag{3}$$

where the relation between the coefficients of the two levels of resolution is given by

$$c_n^j = \sum_k h_{k-2n} c_k^{j-1}\,, \quad d_n^j = \sum_k g_{k-2n} c_k^{j-1} \tag{4}$$

and $g_n = (-1)^n h_{-n+1}$. h and g are the low-pass and high-pass filters respectively. The decimation by a factor 2 corresponds to a down-sampling when going from one level to the next coarser one. This decomposition can be continued using the relation $V_j = V_{j+1} \oplus W_{j+1}$ and so on until a given level $J > j$, obtaining the following approximation for f:

$$f^{j-1} = \delta^j + \cdots + \delta^{J-1} + \delta^J + f^J$$

The inverse operation, the reconstruction of f^{j-1} from f^j and δ^j, is simply given by:

$$c_k^{j-1} = \sum_n (h_{k-2n} c_n^j + g_{k-2n} d_n^j) \tag{5}$$

Filters for analysis and reconstruction satisfying the relations (4) and (5) are called quadrature mirror filters (QMF). Furthermore, they constitute a subband filtering scheme with exact reconstruction.

2.3 The Problem of Selecting a Basis

Depending on the procedure chosen for the computation of the coefficients h_n which solve the two scale relation (2), it can be shown [Dau92], that for a fixed support width of ϕ, the basis functions obtained can have either more regularity or a larger number of vanishing moments, but not both simultaneously. The number of vanishing moments is N, if

$$\int \psi(x)x^l dx = 0\,;\; l = 0, \cdots, N$$

At this point one has to decide what is more adequate for the problem under consideration: regularity or vanishing moments. If the basis functions have a large number of vanishing moments, they will have good approximation properties which correspond to a higher compression potential, but they will exhibit a lack of regularity. More regularity in the basis functions, however, corresponds to smooth elementary building blocks in the decomposition introducing less perceptible quantization errors in the compression.

3 The Rendering Algorithm

This section describes how to solve the volume rendering integral using a multiresolution representation of the volume density function [GLDK95, Wes94]. In order to straighten out a common misinterpretation of Muraki's [Mur93] work we point out, that we do not project the volume rendering integral operator, which is highly nonlinear, since it is not even obvious, if such a projection is possible. For applications of orthonormal wavelets in non linear analysis see for example [Jaf93].

The rendering algorithm based on the pyramidal representation of the volume data consists of the following steps:

- Project the volume density function f into a new coordinate system, in our case a wavelet bases, and then compress it.
- Use this projection as an interpolation scheme to approximate the value of f at any point x in space.
- Solve the volume rendering integral using a standard volume ray casting algorithm.

With respect to the first step of the decomposition of the volume f into a pyramidal representation we can assume that, by means of appropriate rescaling of f, the finest level of resolution, at which the function is *known*, is $j = 0$. We compute all the wavelet coefficients $< f, \psi_{j,n} >, j \geq 0$ by applying the filtering scheme (4). The *scaling* expansion of f after J decomposition steps is given by

$$f(x) \sim \sum_{n \epsilon Z} < f, \phi_{J,n} > \phi_{J,n}(x) + \sum_{j=1}^{J} \sum_{n \epsilon Z} < f, \psi_{j,n} > \psi_{j,n}(x) \tag{6}$$

In order to compute the scalar products $c_n^0 =< f, \phi_{0,n} >$, a quadrature formula for the evaluation of integrals of the wavelets has to be used [SP94]. Due to efficiency considerations, we use a one point formula. If $Q[f(x)]$ is the quadrature formula and $x_1 = M_1$, with M_1 being the first moment of the scaling function, then we have

$$c_n^0 \sim Q[f(x)] = f(x_1)$$

This formula has degree of accuracy 2 for scaling functions with $N > 1$, where N is the number of vanishing moments of the corresponding wavelets [SP94], producing correct results for $N \leq 3$. By projecting f into the *approximation space* V_0 and then applying the discrete wavelet transform J times, we have a new coordinate representation of our data. Depending on the approximation properties of the wavelets many of these coordinates, i.e. basis functions, can be neglected. In order to discard these coefficients we use a hard threshold policy, where all coefficients which are smaller in absolute value than the threshold are set to zero [CMQW89, Don93].

The second step in the rendering algorithm consists of the evaluation of f at a point in space using the wavelet decomposition. Starting at the finest level of resolution we compute the contribution of all non-zero wavelet coefficients and add them to the contribution corresponding to the scaling function. Within this loop we check for the finest level at which no wavelet coefficient contributes to f. If a level $j > 1$ ($j = 1$ being the finest level) is found, the integration step size will be increased by a factor of 2^{j-1}, thus adapting the integration step size to the information contained in the wavelet coefficients. Even for a wavelet basis with many vanishing coefficient, i.e. with many wavelet coefficients set to zero after thresholding, the traversal of this pyramidal data structure is quite time consuming. In the worst case, where all the wavelet coefficients are different from zero, about one hundred floating point multiplications are needed for interpolating only one point using basis functions of support width two and three decomposition steps. If the support width of the wavelets used is four more that seven hundred floating point multiplications are needed.

The last step of the rendering algorithm consists of the evaluation of the volume rendering integral, where any standard volume ray casting algorithm can be used. All known techniques for improving performance can still be exploited, such as β-acceleration, frequency cutting, space leaping and early ray termination. The one new aspect here is the way in which the integration step size is adapted by using the information contained in the wavelet coefficients.

4 Sparse 3D Arrays

The last section showed that the evaluation of the pyramidal data structures representing the volume density function in a wavelet bases is intrinsically a very compute intensive task. In order

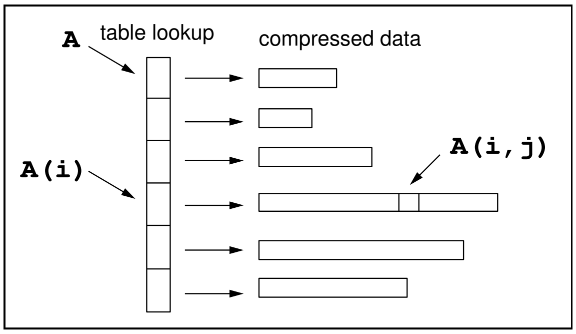

Figure 1: Schematic representation of a 2D sparse array

to improve performance we have to optimize each one of the involved processing steps. Looking at standard methods for data compression, which are designed to obtain the maximum compression possible, one finds that the access rate to a data point, i.e. obtaining the value of a wavelet coefficient from the compressed data, is one of the most time consuming steps during the whole rendering process. We first briefly present one of these standard algorithms [Wes94]. For each level of resolution in the decomposition hierarchy the following steps are carried out:

- normalization: normalize all coefficients to the interval $[0, 1]$ and discard those that are smaller than a certain threshold (e.g. between 0.05 and 0.2).
- quantization: decide how many bits will be used to represent data at the coarsest level, then increment this quantization one bit per level in order to get a better resolution at finer scales. Typical applications use four bits.
- compression: all seven wavelet coefficients that correspond to an index triple (i, j, k) are grouped with a multi stage run-length coding eliminating all zero entries.

After encoding the result is a long bit sequence which contains all the necessary information to obtain a data value for a given index triple. The disadvantage of this procedure is now obvious: if one wants the value of a wavelet coefficient, one has to search through this bit sequence always starting at the beginning. For values which are at the very end of the sequence this procedure is very slow.

In order to overcome this problem we have implemented our data structures using techniques which are well known in the finite element community. The main idea behind the compression method is to avoid starting from the very beginning of the compressed bit sequence when searching for the value of a wavelet coefficient, but to encode the address of the first two indices of the index triple. The seven wavelet components corresponding to a level are saved independently in a 3D scalar array, and are independently compressed according to the following scheme:

- normalization and thresholding: these steps are implemented as described above.
- quantization: to represent our data we use eight bits, since byte operations are faster and can be optimized. At the coarser levels we do require more memory for each coefficient than using the standard scheme, but we recall that at these levels we actually have fewer coefficients, due to the decimation in the filtering process.
- compression: all coefficients corresponding to the z direction, i.e. for an index triple (i, j, k) all coefficients corresponding to the k index for fixed i, j, are run-length encoded.

- table lookup: the resulting bytes sequences are arranged in a lookup table which corresponds to the i, j position in the 3D array, an example for the 2D case can be seen in figure 1. Thus, the access to the begin of a byte sequence is very fast. Then, the search for a value is carried out sequentially. The size of this lookup table is small compared to the total size of the scalar field being compressed. For a data set of size 256^3 its size is 256 KBytes.
- caching: the address corresponding to the last k index is cached. This works well for many applications, where large loops are processed. One can always arrange these loops such that k is the faster index. In this case only one operation is needed to access the next data value.

5 Results and conclusions

In this paper we have proposed a new data structure for compressed scalar fields in order to achieve better volume rendering performance. The highest compression rates that can be achieved are limited by a lookup table introduced in order to accelerate the access to the wavelet coefficients. In order to demonstrate the advantages of this method compared to other algorithms for volume rendering on compressed data we consider three different aspects: the access rate to the compressed fields, the compression rate, and the rendering times compared to those for uncompressed data. We have used two data sets with different features in order to test wavelet bases with different properties, i.e. a different number of vanishing moments. The first data set was generated with a function which is C^∞ everywhere but in a region where we have introduced a discontinuity. The second data set is a standard MRI of a human head with a size of 128×256^2. We have used the Haar-Wavelet and the Daubechies wavelets with 2 and 3 vanishing moments.

In a first step we compared the cumulative time for accessing each element within an uncompressed 3D array to the time accessing each element within a compressed volume using the algorithms and data structures described above. From table 1 we can see that the enormous overhead introduced is still noticeable for a compression threshold of 0.05. For a value of 0.2, however, the overall access time to the compressed data set is already faster than the standard access to an uncompressed 3D field. Obviously, using the Daubechies basis functions with a larger number of vanishing moments even higher compression rates and correspondingly shorter access times can be achieved.

threshold	Haar	Daubechies4	Daubechies6
0.05	1.64	1.40	1.15
0.20	0.84	0.64	0.64

Table 1: Relative access rates for compressed data using different wavelet bases

Data compression is one of the fields where the discrete wavelet transform is most successfully applied. The compression rates that can be achieved depend on the data properties and the wavelets used. For standard basis functions, such as the Daubechies wavelets, and standard applications, where the images exhibit a strong variation in the pixel values, compression rates of $k = 50$ can be reached while maintaining good quality [LMR94]. From table 2 we conclude that for the large data set corresponding to the human head we achieve a compression rate of 43 to 45 with three decomposition steps using the Daubechies wavelet with 2 respectively 3 vanishing moments and a threshold of 0.2. Figures 2 to 4 show that even for the large threshold of 0.2 good (considering the image size of 200 x 200) image quality can be obtained. Obviously, better compressions rates for the same image quality can be achieved using wavelets with more vanishing moments, due to the better approximation properties of these basis functions. Furthermore, as can be seen in figures 5 to 7 the effects of the compression are much more visible in the case of the smooth function than in the case of the human head data set, since the wavelet functions used are not well adapted for approximating functions with such smooth variation in their values.

Dataset	size	threshold	Haar	Daubechies4	Daubechies6
smooth	1MByte	0,05	43.6 KByte	40.9 KByte	40.8 KByte
smooth	1MByte	0,2	38.9 KByte	38.5 KByte	38.4 KByte
head	32 MByte	0.05	1695.5 KByte	1544.0 KByte	1436.4 KByte
head	32 MByte	0.2	791.7 KByte	732.3 KByte	703.6 KByte

Table 2: Compression results for two datasets using different wavelet bases

The implementation of the algorithm was performed within the VolVis volume rendering package [AHH+94] which can be obtained in source code from `volvis@cs.sunysb.edu`. Some of the VolVis routines had to be modified in order to include the adaptive integration step algorithm and the different interpolation schemes introduced above. However, the graphical user interface, the input/output routines and the basic raytracing kernel of VolVis remained unchanged providing the same rich functionality.

threshold	smooth func.	3D-head
0.05	0.66	0.40
0.1	0.55	0.25
0.2	0.43	0.15

Table 3: Speedup in rendering time for various compression thresholds

Based on three decomposition steps and using the Daubechies wavelets with 2 vanishing moments we compared the rendering time corresponding to different threshold values with the rendering time using the same algorithm on uncompressed data. The results shown in table 3 demonstrate that significant improvements can be achieved when rendering compressed data. However, due to the overhead resulting from the wavelet-based interpolation, the standard VolVis rendering times can not be achieved, yet. This makes the proposed volume rendering method suitable for environments where a significant decrease in memory requirements while maintaining good image quality outweighs the moderate increase in rendering time.

References

[AHH+94] R. Avila, T. He, L. Hong, A. Kaufman, H. Pfister, C. Silva, L. Sobierajski, and S. Wang. Volvis: A diversified volume visualization system. In R. D. Bergeron and A. Kaufman, editors, *Visualization '94*. IEEE Computer Society Press, Los Alamos, California, 1994.

[Chu92] C. K. Chui. *An Introduction to Wavelets*. Academic Press, San Diego, 1992.

[CMQW89] R. R. Coifman, Y. Meyer, S. Quake, and M. V. Wickerhauser. Signal processing and compression with wave packets. In Y. Meyer, editor, *Proceedings of the International Conference on Wavelets, Marseille*. Masson, Paris, 1989.

[Dau92] Daubechies, Ingrid. *Ten Lectures on Wavelets*. Number 61 in CBMS-NSF Series in Applied Mathematics. SIAM, Philadelphia, 1992.

Figure 2: Threshold 0.0

Figure 3: Threshold 0.1

Figure 4: Threshold 0.2

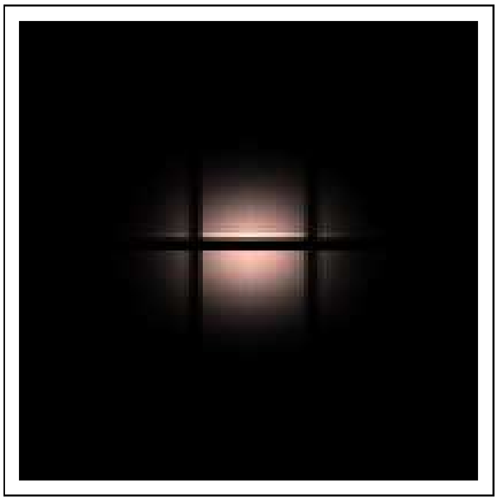

Figure 5: Threshold 0.0

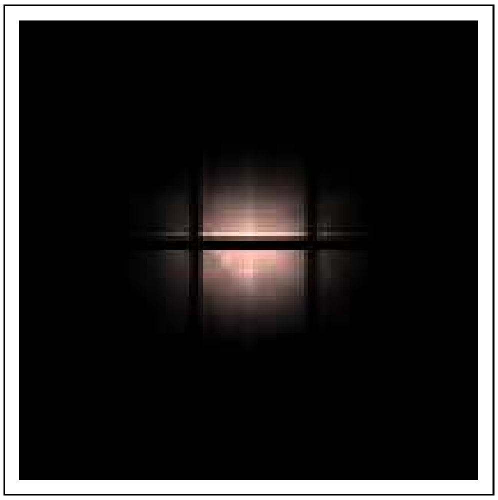

Figure 6: Threshold 0.1

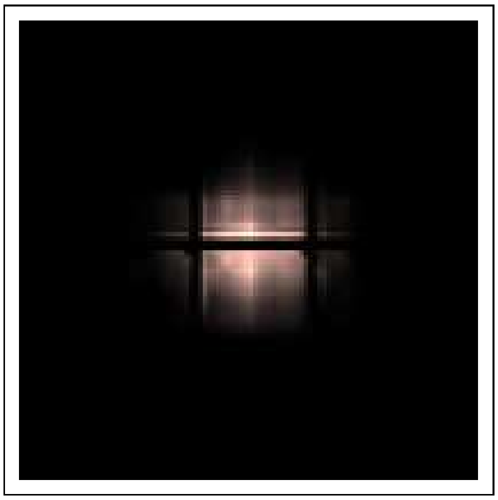

Figure 7: Threshold 0.2

Images volume rendered with the Daubechies wavelet with 2 vanishing moments

[Don93] D. L. Donoho. Unconditional Bases are Optimal Bases for Data Compression and for Statistical Estimation. *ACHA*, 1(1):100–115, 1993.

[GLDK95] 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. *Comput. & Graphics*, 19(1):47–62, 1995.

[Jaf93] S. Jaffard. Wavelets and nonlinear analysis. In J. Benedetto and M. Frazier, editors, *Wavelets: Mathematics and Applications*, pages 467–503. CRC Press, Boca Raton, FL, 1993.

[Kru91] W. Krueger. The Application of Transport Theory to the Visualization of 3-D Scalar Data Fields. *Computers in Physics*, 5(4):397–406, April 1991.

[LMR94] A. K. Louis, P. Maass, and A. Rieder. *Wavelets.* B. G. Teubner Stuttgart, Germany, 1994.

[Mur93] S. Muraki. Volume Data and Wavelet Transforms. *IEEE Computer Graphics and Applications*, 13(4):50–56, 1993.

[PD84] T. Porter and T. Duff. Compositing digital images. *Computer Graphics*, 18(3):253–259, July 1984.

[Sab88] P.A. Sabella. A Rendering Algorithm for Visualizing 3D Scalar Fields. *Computer Graphics*, 22(4):51–58, August 1988.

[SP94] W. Sweldens and R. Piessens. Quadrature Formulae and Asymptotic Error Expansions for wavelet approximations of smooth functions. *SIAM J. Numer. Anal.*, 31:1240–1264, 1994.

[Wes94] Rüdiger Westermann. A Multiresolution Framework for Volume Rendering. In A. Kaufman and W. Krüger, editors, *1994 Symposium on Volume Visualization*, pages 51–58. ACM SIGGRAPH, 1994.