

A Comparative Study of Mipmapping Techniques for Interactive Volume Visualization

Sebastian Maisch
Ulm University
sebastian.maisch@uni-ulm.de

Robin Skånberg
Ulm University
robin.skånberg@uni-ulm.de

Timo Ropinski
Ulm University
timo.ropinski@uni-ulm.de

Abstract

Many large scale volume visualization techniques are based on partitioning the data into bricks, which are stored and rendered using mipmaps. To generate such mipmaps, in most cases an averaging is applied such that an area in a lower mipmap level is presented by the areas' average in the next higher mipmap level. Unfortunately, this averaging results in the fact that mipmaps are not feature-preserving, as details are often lost. In this paper, we discuss and compare mipmap modification schemes which have been developed to support feature-preserving reconstruction during rendering. In particular, we focus on reconstruction schemes which are capable to support anisotropic and non-linear reconstruction, as these are promising to preserve features that are often sacrificed by averaging. The presented techniques are discussed in detail and are thoroughly compared in a quantitative and qualitative analysis. We will discuss their impact on performance, memory footprint and visual quality with respect to feature preservation. Based on the findings we present guidelines for generating and using mipmaps in various visualization scenarios.

Keywords

Volume Rendering, Mipmapping

1 INTRODUCTION

Large data in volume visualization is a very common case in modern visualization [1]. With current methods of data acquisition the resolution of volume data gets too big to be handled in a straightforward way, even on modern graphics hardware. To be able to handle these amounts of data, multi-resolution techniques are often used. Besides plain bricking, most of these techniques rely on downsampling at least parts of the volume to achieve lower resolutions in more distant parts of the volume [2]. This results in a smaller memory footprint of the data to be rendered. Although we will compare different three dimensional downsampling strategies towards mipmapping, our results can also be applied to other approaches in large volume rendering, especially multi-resolution techniques. Mipmaps were originally introduced in computer graphics for two dimensional textures [3] and have later also been extended to volumetric data sets (e.g., [4]). Volume visualization benefits from mipmapping, as it reduces aliasing problems and at the same time lowers the amount of memory necessary for rendering.

While most modern volume visualization algorithms rely on mipmaps (or similar techniques for data downsampling) to support large data [2], mipmaps do not

preserve the underlying data very well and thus details are often lost. We identify two main reasons for this loss: the representation of the entire data range of a cell to a single average value, and the disregard of directional information during the averaging process. Due to the importance of mipmapping in large-scale volume visualization, we investigate and analyze mipmapping modifications, which tackle these shortcomings. We focus on those approaches which do maintain a low memory footprint and enable a direct interactive transfer function change, without requiring extra computations. We see those two criteria as essential capabilities for a technique which shall be scalable to large-scale data. Furthermore, as mipmapping can be considered sufficient for the first downscaling level, where linear interpolation would be used on the GPU, we primarily focus on higher levels, where averaging forbids feature-preservation in the final visualization.

In this paper, we compare two groups of mipmapping techniques in order to investigate their impact on the averaging and directional issues as mentioned above. The first method is named '*Non-Linear Reconstruction*' and uses an enhancement of mipmaps that not only stores the average value of each lower resolution voxel but also minimum and maximum values of the covered area. During rendering, we use these values to reconstruct a power function inside each voxel that tries to resemble the intensity distribution.

With '*Anisotropic Mipmaps*' we try to encode directional information in lower resolution volumes to be able to better reconstruct anisotropy inside each voxel. We use piecewise linear functions to achieve this which are also evaluated during rendering. For each of these techniques we compare different variants with respect to performance, memory footprint and image quality.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.


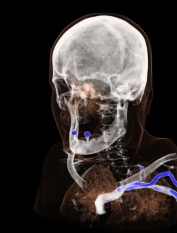
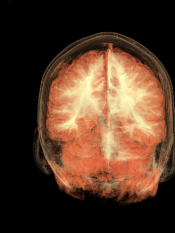
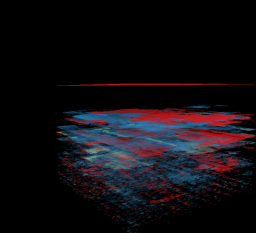
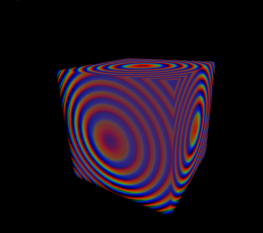
				
Aneurysm (256 ³) Medical (CT)	Male (512 ² × 460) Medical (CT)	Brain (128 × 256 × 109) Medical (MRI)	SEG Y (256 ³) Seismic	Spherical (256 ³) Synthetic

Table 1: Overview of the datasets used throughout the paper; the name, dimensions, and acquisition modality are listed. The images show renderings of the highest resolution with the respective transfer functions applied.

2 RELATED WORK

Today, several different approaches exist to support rendering of multi-resolution volumes. On a data structure level, octrees and kd-trees are widely used techniques to handle large-scale volumetric data sets [2, 5]. Other techniques use hierarchical grids to represent the data [6, 7]. In both cases mipmaps [3] are commonly used for storing and downsampling the volume.

In the recent past several approaches have been put forward to improve the visual quality of mipmapping by trying to reconstruct the intensity distribution of the region covered by the downsampled voxel. Such approaches are based on Gaussians [8] or sparse PDFs [9, 10]. The latter group of techniques uses a hierarchical representation of the data making use of Gaussians to resemble the data range. This comes at the need of performing calculations with every change in transfer function. Independent of the method used to store the intensity distribution, the memory footprint of these methods is larger than with plain mipmaps. Therefore, we have omitted these approaches from our comparison, as we aim for a low memory footprint while still being able to edit the transfer function during rendering without a significant computational overhead. Instead, we investigate an improved aggregation technique which is similar to the Min-/Maxmaps used for 2D shadow mapping by Guennebaud et al. [11] and for storing minimum and maximum positions of geometry in Mipmaps by Carr et al. [12]. Thus, to reconstruct the data distribution of each voxel, we use a volume texture storing for each voxel the minimum, maximum, and average values of the region they cover. We found this relevant, because the usage of minimum and maximum values in volume rendering is a frequently used approach. Lacroute and Levoy [13] for instance introduced Min-Max Octrees in which each octree node stores a minimum and maximum value of the contained volume to allow empty space skipping. A similar concept is used by Dong et al. [14]. As many of the features suppressed in standard mipmapping have a directional nature, we also look into encoding the change of intensity within a downsampled voxel by using a technique similar to deep shadow mapping [15]. Thus, anisotropic features can be captured with anisotropic voxels [16, 17].

Kraus and Ertl [18] introduced an approach to downsampling that tried to preserve the topology of isosur-

faces. Their memory requirements are the same as conventional mipmapping, but there are cases where it is not possible to preserve the topology, for example, if there are more than one local minima or maxima in the original resolution data that corresponds to a single downsampled voxel. Later Kraus and Ertl have addressed a similar issue for downsampling RGBA volume data [19]. We have not included these approaches in our evaluation, as the first one is intended for extracting isosurfaces from a downsampled volume. While some ideas as the need to preserve extreme intensity values also apply to raycasting volumes this approach is not entirely suitable. The latter technique is only applicable to volumes that evaluate the transfer function as a preprocessing step which we want to avoid, because of the inability to change the transfer function without recomputing the volume.

While we will not focus on compression techniques, we would like to mention that they can be combined with the tested techniques to further reduce the required memory footprint. Often, for instance, wavelet representations are used to compress large volumes and thus reduce the loss of information [20, 21]. These or other techniques, may also be combined with the approaches discussed in this paper.

3 METHODOLOGY

The evaluation presented in this paper is motivated by two common problems that arise when using mipmaps. One problem that is also addressed in other papers is the loss of information about the intensity distribution due to averaging the voxels intensity values. We use a simple non-linear reconstruction approach to address this issue and show, that even under very harsh conditions and high frequency transfer functions, we can provide convincing results. We have selected the method described in Section 4.1 among the known techniques addressing similar issues as it is simple to realize and can be directly integrated into existing volume renderers. Furthermore, it fulfills our requirements with respect to low memory footprint and the flexibility of post-classification. Kraus and Ertl [18] point out the importance of preserving the topology of isosurfaces within the volume. This includes the ability to reconstruct the extreme values for each downsampled voxel. We do not target the generation of isosurfaces, but the importance of keeping the extreme values for a better reconstruction of intensity values is still applicable.

The second problem when using mipmaps is that directional information is lost, also due to the averaging which occurs in the process of generating the mipmaps. To deal with this issue, we aim at evaluating how the encoding of directional intensity changes can improve feature preservation. The idea is to derive a more accurate intensity value from this directional information, to which we then apply the transfer function during rendering. The underlying method for realizing this encoding is described in Section 4.2.

In this paper, the i -th mipmap level is specified using the convention ℓ_i where ℓ_0 represents the full resolution data. For each subsequent level ℓ_{i+1} the resolution is halved compared to the previous level ℓ_i .

By taking the two exemplary techniques into account, we want to evaluate the impact of non-linear reconstruction vs. anisotropic reconstruction, when using mipmaps in volume rendering. Based on our own observations as well as the results reported in recent papers, e.g., [10], we consider the mipmap levels ℓ_0 and ℓ_1 (the highest and second highest resolution) sufficient in terms of quality of the resulting image, and therefore focus on the higher levels. The representative approaches are implemented inside a basic raycaster for volume data which incorporates early ray termination.

Our testing environment contains five different data sets (see Table 1). We use several volumes from the medical domain. We also include a seismic data set, and one synthesized by using a simple spherical function (as was also used by Younesy et al. [8]). These different volumes represent a wide range of different applications. Transfer function design is discussed in Appendix A.

The measurements we used to determine the quality of each technique in the different scenes are as follows. We measure memory footprint and performance compared to rendering a full resolution volume and the original mipmapping. We also take into account the error rates (PSNR) of the images in comparison to an image produced using the highest resolution data as a ground truth. In terms of quality we will look at different features of the datasets that are visible if we render the high resolution data and discuss how well these are preserved in downscaled renderings.

4 EVALUATED RECONSTRUCTION TECHNIQUES

In this section, we describe the two techniques we use to improve reconstruction of downsampled values in different situations. One technique aims to preserve the intensity distribution of each downsampled voxel using a non-linear function. We will call this technique ‘non-linear reconstruction’. The other approach is to preserve a directional distribution of values in downsampled voxels. This technique will be called ‘anisotropic reconstruction’.

4.1 Non-Linear Reconstruction

To reconstruct the intensity distribution of a downsampled voxel in a non-linear manner, we want to find a

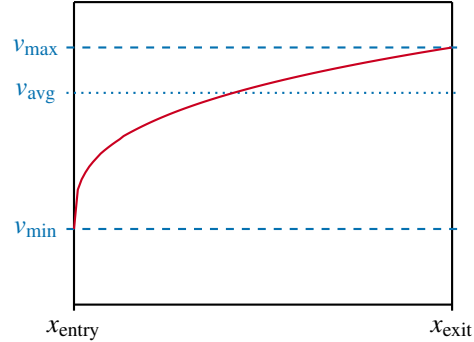


Figure 1: Reconstruction function (*red*) inside a voxel defined by the ray entry (x_{entry}) end exit (x_{exit}) points and the minimum (v_{min}), maximum (v_{max}) and average (v_{avg}) values of the covered high resolution voxels in ℓ_0 .

simple representation of that distribution encoded using low memory footprint. Using only three values – the minimum, maximum, and average in a specified region, we fit a power function to these values covering the interval between the minimum and the maximum value while still preserving the average. To be able to create such a representation during rendering we need to preprocess the volume data and store for each downsampled voxel the minimum (v_{min}), maximum (v_{max}), and average (v_{avg}) value of that voxel’s equivalent region in the original volume.

When ray-casting the volume represented by ℓ_2 or lower we use these values to reconstruct a non-linear function representing the intensity distribution inside each voxel. Younesy et al. [8] use a Gaussian to reconstruct this value from the mean and standard deviation values they store. They then use a preintegrated transfer function to calculate the color value at each voxel. Our method works without any precalculations directly using the transfer function. To generate the final color value we sample the reconstructed function, apply the transfer function to each value, and composite the colors using the scheme determined by the raycasting algorithm. Therefore the reconstructed function is interpreted as a one dimensional function along the ray used to calculate the current pixels color. This introduces an error as we have no actual information about the locations of each value contained in the intensity distribution, but imply these locations by using our reconstruction. However, as Kraus and Ertl have discovered [18], this omission of the actual sample orders can be in most cases neglected. The function we use for reconstruction, is a scaled power function:

$$f(x) = (v_{\text{max}} - v_{\text{min}}) \left(\frac{x - x_{\text{entry}}}{x_{\text{exit}} - x_{\text{entry}}} \right)^a + v_{\text{min}} \quad , \quad (1)$$

where x_{entry} is the rays entry point into the voxel and x_{exit} is its exit point. The parameter a is determined by the voxels average value (v_{avg}) to ensure the average of $f(x)$ matches that of the voxel. Figure 1 illustrates this function and the parameters used for its reconstruct-

tion. This parameter is calculated on the fly during ray-casting by solving the definition of the average value:

$$v_{\text{avg}} = \frac{1}{x_{\text{exit}} - x_{\text{entry}}} \int_{x_{\text{entry}}}^{x_{\text{exit}}} f(x) dx \quad , \quad (2)$$

which gives us a as follows:

$$a = \frac{v_{\text{max}} - v_{\text{min}}}{v_{\text{avg}} - v_{\text{min}}} - 1 \quad (3)$$

The impact of minimum, maximum and average value on the shape of the function is illustrated in Figure 1.

The resulting function is then evaluated at different positions within the voxel and the transfer function is applied to these results. The final values are then composited in a straightforward way by using front to back composition. To further improve the coverage of the intensity distribution inside the voxel we adapt the step size we use to fit the function. The same step size is used for the composition. In general we use values that are evenly distributed across the intensity range, whereby we use the minimum and maximum values to determine the possible range of values. When sampling the function at N different positions inside the voxel we use the values v_i ($i = 0 \dots N - 1$) defined as $v_i = \frac{i}{N-1}(v_{\text{max}} - v_{\text{min}}) + v_{\text{min}}$ and calculate the step sizes needed to correctly weight those values during composition. The corresponding sampling points for the values v_i are x_i defined as:

$$x_i = \sqrt[a]{\frac{i}{N-1}} (x_{\text{exit}} - x_{\text{entry}}) + x_{\text{entry}} \quad (4)$$

With these points we use the following equation to calculate the step sizes (Δ_i) for ray-casting by using the midpoints between two sampling points to separate steps:

$$\Delta_i = \min \left(\frac{x_{i+1} + x_i}{2}, x_{\text{entry}} \right) - \max \left(\frac{x_i + x_{i-1}}{2}, x_{\text{exit}} \right) \quad (5)$$

Figure 2 illustrates the different v_i , x_i and Δ_i determined by a given function $f(x)$.

To directly see the impact of the non-linear density distribution reconstruction, we will also evaluate the same approach, whereby we replace the non-linear with a linear function. This approach only uses the minimum and maximum values, and we will refer to it as ‘*Linear Reconstruction*’ throughout this paper.

4.2 Anisotropic Reconstruction

The second identified downside of plain mipmapping is the fact, that directional features are not preserved. To deal with this issue, intensity changes for selected

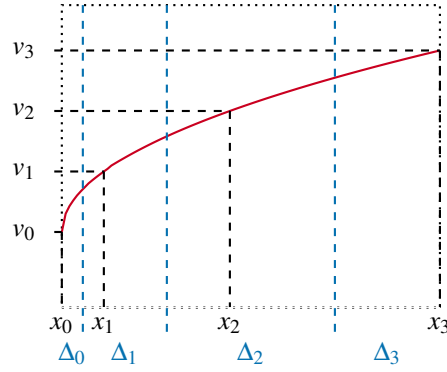


Figure 2: Step sizes (blue) for raycasting a reconstructed function (red) with $N = 4$ sampling points. The sampled values v_i are shown with their corresponding sampling positions x_i .

directions in each region would need to be encoded. This idea forms the basis for the evaluated anisotropic approach, where we – for each major axis within a downsampled voxel – encode how the intensity changes along the axis, and store this information as a piecewise linear function. This approach requires an explicit pre-processing step where intermediate axis functions are computed and later encoded into piecewise linear functions. These functions can later in the rendering part be sampled and composited into intensity values. In the following three subsections, we discuss how to compute and encode the axis functions, as well as how to combine values of the three piecewise linear functions into one representative intensity value.

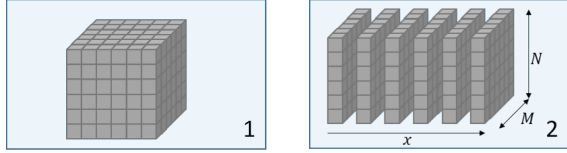
4.2.1 Computing Axis Functions

To compute the axis functions, we begin by studying each of the major axes of a voxel which we intend to downsample (Fig. 3a). The three axes are split into stacks of 2D slices (Fig. 3b), and the mean of each slice is calculated and stored to represent the overall intensity at this location (Fig. 3c). This is always done at the highest possible resolution inside of the region of voxels to capture as many details as possible of the underlying data.

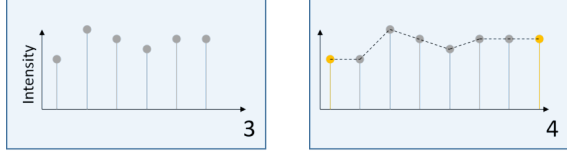
4.2.2 Encoding Axis Functions

With the axis functions represented as lists of N intensity values (Fig. 3c), we aim to compress these using a piecewise linear function *PWLF*. In our case, a piecewise linear function is a function that sparsely stores a set of 2D sample-points (x, y) and the function value can be evaluated at any position x using linear interpolation between the neighbouring points of x . In order to encode the list of intensity values into a *PWLF*, we employ the following greedy algorithm:

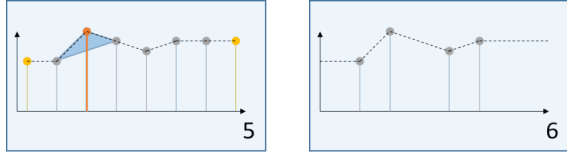
1. We initialize the *PWLF* (Fig. 3d) by adding the full set of axis intensity values with their corresponding positional values along the axis.



(a) The targeted voxel and its data that is about to undergo downsampling. (b) The data is divided along the axis into slices containing $M \times N$ samples.



(c) The mean intensity of each slice is computed and stored with its local position x within the voxel and is padded with copies (yellow) as an intermediate list of pairwise points. (d) The pairs of $(x, \text{intensity})$ are added to a piecewise linear function and is padded with copies (yellow) of the first and last points to aid the computation in the next step.



(e) In an iterative process, the point within the set that has the lowest cost is removed. The cost is calculated as the triangular area between its neighbouring points. (f) The process is stopped when a targeted number of points has been reached or the error of removing the next point above a certain threshold.

Figure 3: The preprocessing step of computing a piecewise linear function that approximates the intensity values along one major axis.

2. We then remove the point in the set which is deemed as the least destructive to overall shape of the function using a cost metric (Fig. 3e). The cost of removing a point is equal to the triangular area which spans the point and its two closest neighbours before and after removal of this point.
3. Step 2 is repeated until we are left with a desired amount of points (Fig. 3f).

4.2.3 Sub-sampling Anisotropic Voxels

When traversing the voxels a sub-sampling approach is used to reconstruct the intensity values within the voxel using the piecewise linear functions stored within it (see Fig. 4). The input arguments for the $PWLFs$ are the local coordinates $(x, y, z \in [0, 1])$ of the sub-sample within the frame of the voxel. The intensity values (Eq. 7) returned by the three $PWLFs$ are then combined into one single intensity value using the dot product of the squared viewing vector \mathbf{d} and intensity values $v(\mathbf{x})$ as shown in Eq. 6

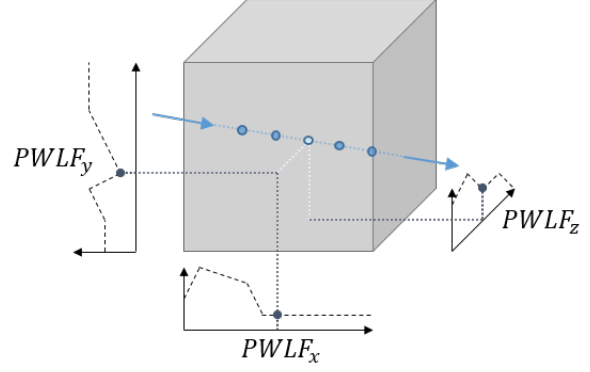


Figure 4: A sub-sample (Light-blue dot) along the traversed ray (Blue) is being composited into a representative value of the intensity within a voxel by using the piecewise linear functions that provide the directional in intensity along each major axis.

$$\text{intensity}(\mathbf{x}) = \text{dot}(\mathbf{d} \odot \mathbf{d}, v(\mathbf{x})) \quad , \quad (6)$$

where

$$v(\mathbf{x}) = \{PWLF_x(x_x), PWLF_y(x_y), PWLF_z(x_z)\} \quad . \quad (7)$$

The major downside of this technique is that it does not allow trilinear interpolation. If trilinear interpolation is used, the resulting intensity values from all of the piecewise linear functions will be averaged over the neighbouring region of the voxel thus resulting in an average intensity, almost indistinguishable from plain mipmapping. This forces us to only use values from within the traversed voxel disregarding the neighbourhood of it. The visual impact of this is that the boundaries of the voxels can be seen in most of the cases.

5 EVALUATION

In this section we analyze the impact of the mipmapping shortcomings resulting from the area averaging and the non-directional reconstruction. We do so by comparing standard mipmapping against non-linear and anisotropic reconstruction, as they have been described in the previous section. The achieved results are evaluated quantitatively and qualitatively. A more comprehensive list of results can be found in Appendix B.

5.1 Quantitative Evaluation

The presented techniques are meant to produce better images than standard mipmapping while still using less GPU memory and being faster as compared to rendering the highest resolution data. For performance measurements we ray-casted the volume data with a screen resolution of 1280×720 with the volume fitting roughly to the screen, whereby we have used the same images in the subsequent error analysis. The results of the performance evaluation, as summarized in Table 2, have been measured on a 3.60 GHz Intel i7 Haswell system with 16GB RAM and a NVIDIA GTX 980 GPU.

As it can be seen in Table 2, the results of non-linear reconstruction highly depends on the size of the volume.

Dataset	Non-Linear	Anisotropic	Mipmaps
Aneurysm (ℓ_2)	76%	60%	122%
Male (ℓ_3)	232%	51%	385%
Brain (ℓ_2)	69%	16%	109%
Spherical (ℓ_3)	138%	90%	207%
SEGY (ℓ_2)	102%	16%	150%

Table 2: Performance measurements of different techniques relative to rendering at the original resolution (ℓ_0). A value of 200% signifies that the technique is twice as fast as the reference, while a value of 50% signifies that only half the speed of the reference was achieved.

For volumes that use a lot of memory (see Table 1) in the first place texture fetches are more expensive (due to cache misses) and a reduction of the volumes’ resolution improves the frame rates drastically. For lower resolution volumes this is not the case, and the computation time overhead for the evaluation of the reconstructed function seems to have more impact. The rendering performance of the ‘Spherical’ data set is in general rather high, as the chosen transfer function makes the volume appear very dense which triggers the early ray termination.

The performance results of the anisotropic voxels seem worse than rendering the high resolution data, this is due to a bigger number of texture fetches which are needed to retrieve the data of the piecewise linear functions. The memory requirements for the anisotropic voxels is noticeably larger than with the other techniques, but has been implemented in a very naïve way where one voxel stores three unique piecewise linear functions. In practice, many of the piecewise linear functions are identical or very similar, which makes them a good candidate for lossless or lossy compression through clustering.

With no compression applied the memory requirements for the tested techniques directly depend on the resolution of the original data. The piecewise linear functions of the anisotropic mipmaps are always represented using eight 8-bit integers per pixel – four for position and four for intensity. The non-linear reconstruction on the other hand uses the same bit-depth as the original volume. Thus, the memory requirements of the reconstruction methods (as well as regular mipmapping) are a fixed percentage of the original memory (118% for non-linear reconstruction, 116% for linear reconstruction and 114% for regular mipmapping), whereas the anisotropic mipmaps use 156% of the original data sets memory for 8-bit volumes. The percentage gets lower for higher bit-depths of the original volume due to the used 8-bit quantization.

The more interesting figures are the error measurements of the rendered final images. The images can be seen in Appendix B. We calculated the Peak Signal to Noise Ratio (PSNR) of the resulting images, by taking into account the images produced with the high resolution data sets as a ground truth. We measured these values

for both, a high resolution (1280×720 Pixels) as well as a low resolution (128×72) image. The high resolution was chosen to be able to clearly see differences and errors introduced by both types of methods. The lower resolution represents the actual size of the images when the size of one voxel in the data set should correspond to the size of one pixel, whereby the mipmap levels have been chosen accordingly. To avoid aliasing we use downsampled versions of the high resolution images. The results are shown in Table 3.

In general, the lower resolution images show better quality with respect to the PSNR because quality shortcomings depending on the resolution are avoided. These differences can be seen best for the anisotropic mipmaps as this technique does not support interpolation and thus produces ‘blocky’ results when rendering them in a resolution higher than they are meant for. When comparing the different reconstruction techniques the non-linear reconstruction provides the best results in nearly all cases. The exception is the Spherical data set where the intensity values are linear with the distance from the center. The linear reconstruction ensures that this linearity is preserved well while the non-linear reconstruction might introduce some errors.

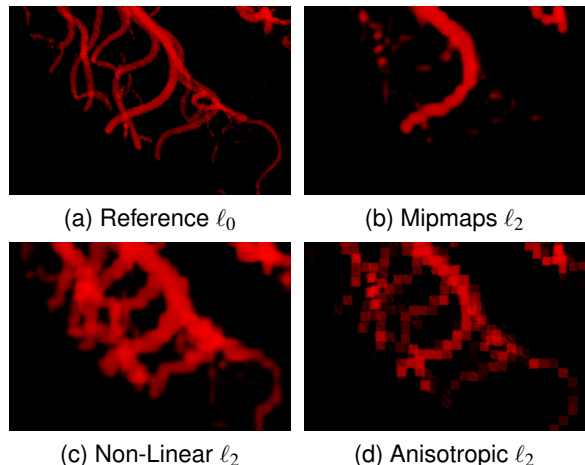


Figure 5: The Aneurysm dataset, rendered at its original resolution ℓ_0 (a) and at resolution ℓ_2 for the downsampled methods. It shows a region with a lot of high frequency details.

5.2 Qualitative Evaluation

By visually comparing the results of the different techniques we can make assumptions about how to address mipmapping problems in general. In Figures 5 and 7 the plain mipmapping clearly shows a loss of information. In Figure 5b the high frequency details are lost due to averaging. The most obvious case is the seismic data set. There nearly all information is lost as can be seen in Figure 7b. When looking at the Spherical data set in Figure 6b most features seem to be preserved. However, when taking a closer look the colors observed seem to be exaggerated and are shifted towards the corners of the volume.

Dataset	Non-Linear		Linear		Anisotropic		Mipmaps	
	high Resolution	low Resolution	high Resolution	low Resolution	high Resolution	low Resolution	high Resolution	low Resolution
Aneurysm (ℓ_2)	20.62 dB	22.18 dB	16.23 dB	17.00 dB	24.49 dB	30.35 dB	23.82 dB	26.95 dB
Male (ℓ_3)	21.02 dB	22.25 dB	18.13 dB	18.84 dB	22.86 dB	24.78 dB	18.96 dB	19.78 dB
Brain (ℓ_2)	26.20 dB	28.33 dB	23.60 dB	24.77 dB	25.05 dB	26.60 dB	23.27 dB	24.79 dB
Spherical (ℓ_3)	23.04 dB	27.45 dB	24.09 dB	28.88 dB	19.93 dB	23.52 dB	22.02 dB	26.04 dB
SEGY (Top) (ℓ_2)	22.23 dB	22.96 dB	21.36 dB	21.98 dB	26.97 dB	32.04 dB	17.69 dB	17.97 dB

Table 3: Error ratios (PSNR) of different techniques, datasets and resolutions.

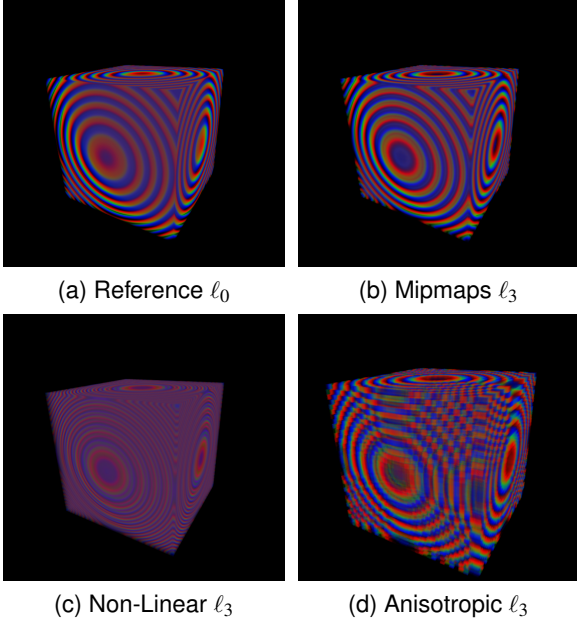
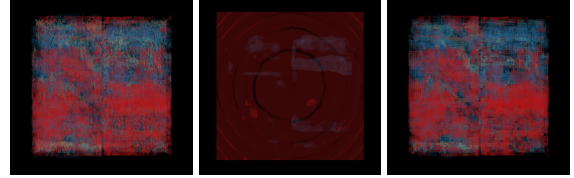


Figure 6: The Sphere dataset, rendered at its original resolution ℓ_0 (a) and at resolution ℓ_3 for the downsampled methods. The fading to grey in Non-Linear reconstruction (c) is an expected artifact to avoid aliasing.

From a qualitative point of view, the anisotropic voxels seem to preserve details better over the plain mipmapping technique, but have the downside of looking ‘blocky’. This ‘blockiness’ comes from the fact that doing trilinear interpolation is too expensive in terms of performance to be practical, and if applied, the intensity is again averaged over the neighbouring region and the result ends up being indistinguishable from mipmapping. A similar problem arises when the different directions need to be interpolated due to a view that is not parallel to the volumes coordinate axes. This can be seen clearly in the geological data set. When choosing a view that is parallel to the volumes up coordinate as can be seen in Figure 7c the results are very similar to the reference image. When dealing with high frequencies as in the Aneurysm data set, anisotropic approaches also generate a good reconstruction of most details as can be observed in Figure 5d. The worst scenario is presented in Figure 6d. Here, preservation of the anisotropy is not important in the final rendering as the volume is very dense. Due



(a) Reference ℓ_0 (b) Mipmaps ℓ_2 (c) Anisotropic ℓ_2
Figure 7: The SEGY dataset, rendered from a top view at its original resolution ℓ_0 (a) and at resolution ℓ_2 for the downsampled methods. Plain mipmapping does not preserve the structure of this dataset. From this perspective the Anisotropic Mipmapping (c) provides results that are very similar to the reference image.

to the interpolation of axis functions the result is very similar to the plain mipmapping.

When studying the non-linear reconstruction approaches, the different regions classified by the transfer function are well preserved. Due to the lower resolution of the volume these areas seem bigger in all cases. This can be seen especially well in Figure 5c where the high frequency details are rendered as bigger structures. This also leads to more saturated colors in nearly all of the results achieved using this approach. Several images show that preserving not only the extreme values but also the average value is important to create convincing results. In nearly all cases the non-linear reconstruction which preserves the average value provides results that are closer to the ground truth. The only exception is the synthetic Spherical data set which only contains intensity values linear in respect to the distance to the volumes center. This explains the close fit that a linear reconstruction provides.

5.3 Guidelines

We have seen different approaches which in general provide better quality compared to plain mipmapping depending on the type of data and the transfer function used. Based on these observations we provide guidelines for using the mentioned approaches in different scenarios.

As mentioned above the non-linear reconstruction exaggerates the color saturation. Generally the regions colored by a transfer function are the ones with features that the observer wishes to emphasize. This may prove useful in cases like seismic visualization to use these or similar approaches to be able to observe all details

in the volume. The same applies to medical datasets where it may be crucial for the observer not to miss any details of significance. In these cases using a non-linear reconstruction will benefit the visualization.

In the cases where you want to preserve high frequencies within data sets, an anisotropic approach may be a viable option. And if the data set contains intensities layered along one major direction, an anisotropic approach may also provide better results.

If rendering speed is the top priority plain mipmaps are the most obvious choice to use due to the native hardware integration existent in all modern GPUs. The Non-Linear Reconstruction provides reasonable speed especially with bigger datasets while providing an improved image quality.

When dealing with large volumes the overall difference in memory between plain mipmaps and techniques better preserving the intensity distribution of downsampled voxels is very small. Using such a technique may be a viable option in those cases.

6 CONCLUSION & FUTURE WORK

We have discussed two shortcomings of plain mipmaps and evaluated techniques that address these using different data sets from different domains together with transfer functions that have been designed to visualize interesting features in these volumes. As a conclusion of the conducted evaluation, we can say that all tested approaches can be implemented as an improvement to plain mipmapping. Also, conserving the intensity distribution of a downsampled voxel looks like a very promising approach that is applicable in the general case. An anisotropic approach also provides promising results but lacks hardware support and a proper solution for feature preserving interpolation in the technique presented in this paper. The different strengths of the provided techniques calls for a combination of both approaches that should provide a reconstruction of the intensity distribution of downsampled voxels as well as a directional component.

In our opinion, the ideal technique for mipmaps in volumetric visualization should fulfill three important requirements. First, it should preserve the intensity distribution of all the data contained within the region of the voxel. Second, it should utilize the GPU's hardware support for interpolation when fetching data. And third, the intensity distribution contained within the voxel should be able to be integrated with the transfer function in an efficient manner enabling interactive editing of the transfer function together with interactive frame rates.

As it seems to be more important to preserve the intensity distribution in a correct way rather than to store the directional intensity changes within a voxel an interesting approach would be to approximate this distribution by using piecewise linear functions. On the other hand the directional interpolation problems of Anisotropic Mipmaps could be tackled encoding directional intensity using spherical harmonics.

REFERENCES

- [1] E. W. Bethel, H. Childs, and C. Hansen, *High Performance Visualization: Enabling Extreme-Scale Scientific Insight*. CRC Press, 2012.
- [2] J. Beyer, M. Hadwiger, and H. Pfister, "State-of-the-Art in GPU-Based Large-Scale Volume Visualization," *CGF*, May 2015.
- [3] L. Williams, "Pyramidal parametrics," *ACM SIGGRAPH Computer Graphics*, vol. 17, no. 3, pp. 1–11, Jul. 1983.
- [4] E. C. La Mar, B. Hamann, and K. I. Joy, "Multiresolution Techniques for Interactive Texture-based Volume Visualization," in *VIS '99*. IEEE Computer Society, Oct. 1999.
- [5] C. Crassin, F. Neyret, S. Lefebvre, and E. Eisemann, "GigaVoxels: Ray-Guided Streaming for Efficient and Detailed Voxel Rendering," in *ISD '09*. ACM Press, Feb. 2009, p. 15.
- [6] M. Hadwiger, J. Beyer, and H. Pfister, "Interactive Volume Exploration of Petascale Microscopy Data Streams Using a Visualization-Driven Virtual Memory Approach," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2285–2294, Dec. 2012.
- [7] T. Fogal, A. Schiewe, and J. Krüger, "An Analysis of Scalable GPU-Based Ray-Guided Volume Rendering," *IEEE Symposium on Large-Scale Data Analysis and Visualization*, vol. 2013, pp. 43–51, Oct. 2013.
- [8] H. Younesy, T. Möller, and H. Carr, "Improving the quality of multi-resolution volume rendering," in *IEEE VGTC '06*. Eurographics Association, May 2006, pp. 251–258.
- [9] M. Hadwiger, R. Sicut, J. Beyer, J. Krüger, and T. Möller, "Sparse PDF maps for non-linear multi-resolution image operations," *ACM TOG*, vol. 31, no. 6, p. 1, Nov. 2012.
- [10] R. Sicut, J. Krüger, T. Möller, and M. Hadwiger, "Sparse PDF Volumes for Consistent Multi-Resolution Volume Rendering," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 12, pp. 2417–26, Dec. 2014.
- [11] G. Guennebaud, L. Barthe, and M. Paulin, "Real-time soft shadow mapping by backprojection," in *EGSR '06*. Eurographics Association, Jun. 2006, pp. 227–234.
- [12] N. A. Carr, J. Hoberock, K. Crane, and J. C. Hart, "Fast GPU ray tracing of dynamic meshes using geometry images," in *GI '06*. Canadian Information Processing Society, Jun. 2006, pp. 203–209.
- [13] P. Lacroute and M. Levoy, "Fast volume rendering using a shear-warp factorization of the viewing transformation," in *SIGGRAPH '94*. ACM Press, Jul. 1994, pp. 451–458.
- [14] F. Dong, M. Krokos, and G. Clapworthy, "Fast Volume Rendering and Data Classification Using Multiresolution in Min-Max Octrees," *CGF*, vol. 19, no. 3, pp. 359–368, Sep. 2000.
- [15] T. Lokovic and E. Veach, "Deep shadow maps," in *SIGGRAPH '00*. ACM Press, 2000, pp. 385–392.
- [16] J. Mitchell, G. McTaggart, and C. Green, "Shading in valve's source engine," in *ACM SIGGRAPH 2006 Courses*. ACM, 2006, pp. 129–142.
- [17] C. Crassin, F. Neyret, M. Sainz, S. Green, and E. Eisemann, "Interactive indirect illumination using voxel cone tracing," *Computer Graphics Forum (Proceedings of Pacific Graphics 2011)*, vol. 30, no. 7, sep 2011.
- [18] M. Kraus and T. Ertl, "Topology-Guided Downsampling," in *Volume Graphics 2001*, K. Mueller and A. E. Kaufman, Eds. Springer Vienna, 2001, pp. 223–234.
- [19] M. Kraus and K. Bürger, "Interpolating and Downsampling RGBA Volume Data," in *VMV*, 2008, pp. 323–332.
- [20] S. Muraki, "Volume data and wavelet transforms," *IEEE Computer Graphics and Applications*, vol. 13, no. 4, pp. 50–56, Jul. 1993.
- [21] R. Westermann, "A multiresolution framework for volume rendering," in *VVS '94*. ACM Press, Oct. 1994, pp. 51–58.