

# Implicit Adaptive Volume Ray-Casting

Christoph Lürig

Roberto Grosso

Thomas Ertl

Computer Graphics Group  
Universität Erlangen-Nürnberg, Germany

## Abstract

In this work we present an acceleration technique for direct volume rendering, that is based on a preprocessing step. This transforms the original uniform grid into an adaptively reduced tetrahedral grid. The basic idea is to exploit the inherent adaptivity information of the reduced grid for the ray integration with importance sampling. An integration step is always performed from cell face to cell face. This way every cell along the ray is sampled and the integration step size correlates with the tetrahedra volume. In contrast to standard integration techniques no extrapolation or higher order integration for error estimation and step size control is necessary. This guarantees extra acceleration and avoids the problem of missing details if the step size becomes too large. Experiments are made to show the quality of the images obtained and the speed up in contrast to standard volume ray-casting techniques.

**keywords:** adaptivity, irregular grids, volume ray-casting

## 1 INTRODUCTION

The integration acceleration technique followed in this work is based on the idea of importance sampling. In volume areas of high variance the integration step size should be shorter than in areas of low variance. This technique has been described by Danskin in [DH92]. This way the relation of approximation error and number of sample points is optimized. To change the step size during integration Danskin uses a pyramidal data structure and the variance value at a certain position to determine the next step size. In contrast our approach is based on an adaptively reduced tetrahedral mesh to represent the information and the step size is implicitly given by the geometry of the cells. The generation of the

adaptively reduced grid is done by a finite element analyzer that computes the *best* approximation of the volume data for a given number of tetrahedra in the sense of the least squares approximation with the  $L_2$  norm. This strategy results in a fine triangulation of areas with high spatial frequency and in a coarse triangulation of areas of low spatial frequency. Our approach exploits this given adaptivity information. As an explanation, we give a two dimensional example. The MRI-slice of a head, that is displayed in Fig. 1, has been analyzed. The resulting triangular mesh is also displayed at the right hand side of the same figure. Of course other reduction techniques, that generate tetrahedral meshes, as those used in Cignoni et al., Lürig et al. [CDFM<sup>+</sup>94, LE96], could also be used to generate the initial data for our ray-casting algorithm.

There are many different techniques for adapting the integration step size for solving the ray integral equation. The approach followed by Westermann [Wes95] uses wavelet coefficients of the wavelet analyzed signal. The decision for the next step size is based on the wavelet coefficients at the last sample point. The problem of this method is, that a reconstruction at every sample point has to be performed and a complex recursive structure has to be traversed.

The basic idea to use a hierarchical data structure for ray-tracing has been published by Fujimoto et al. in [FT86]. They use an octree for ray-tracing scenes with surface descriptions. The octree offers the possibility to jump over large empty space during intersection calculation. The octree is a simple and efficient way to represent a volume, but it can not be fit to a volumetric data set as well as an irregular grid structure. Irregular grids have no constraints about the orientation of the cell faces as the octree has. This is also the main reason for the dominance of irregular grids in the field of numerical computation. Irregular grids for representing volumes have been used by Cignogni et al. in [CDFM<sup>+</sup>94].

Gross et al. [GGS95] used a mixture of quad trees and wavelets to reduce terrain data. The quad trees are used to generate a triangular mesh, and the wavelet coefficients are used to determine whether to descend in the octree or not.

In contrast to the wavelet based approaches for refinement decisions our approach is based on multi-level finite element analysis that produces directly a tetrahedral grid, that approximates the volume. The construction of this grid and the underlying analysis method is described in Grosso et al. [GLE96] in detail. Some more about the theoretical background of this technique can also be found in Bank [Ban96].

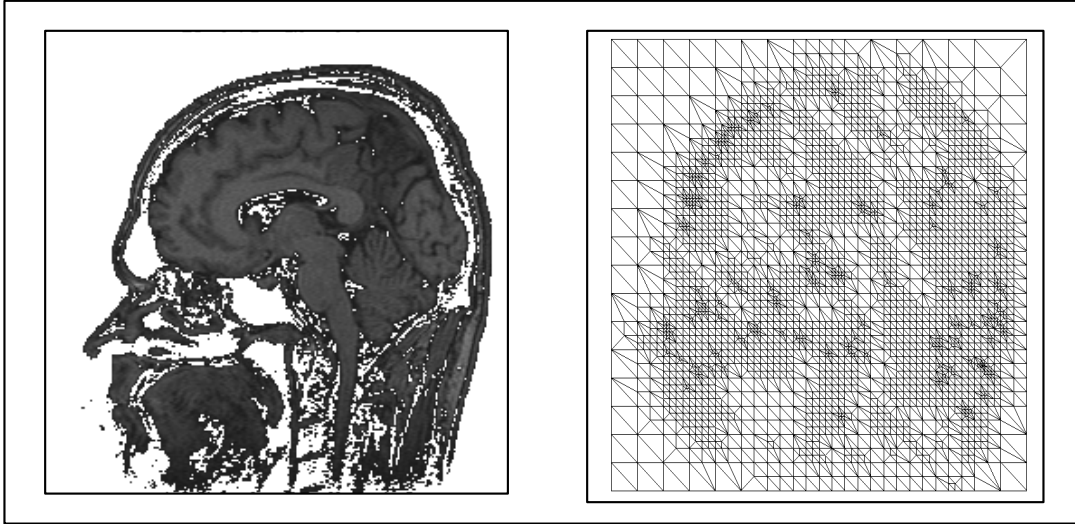


Figure 1: A slice of a MRI-Head and the generated triangular mesh

The basic idea is to start with a coarse triangulation of the volume domain and to refine it afterwards based on a local error estimator for the actual approximation.

In this work we concentrate on the processing of the generated grid. The generated grid contains the position and the values of the vertices, the tetrahedra and the connectivity information of the tetrahedra. This is very crucial for the volume traversal. This differs from the scan-line based algorithm presented by [WVGTG96]. The scan-line based algorithm reduces the cost of determining the next cell and the entrance point as very little resorting during scan-line and scan-plane change is necessary. The algorithm presented in this approach does not need to sort at all, as the connectivity makes it easy to determine the next cell with its entry point. The computation of cell entry and exit point is also described in [Gar90].

Our rendering tool uses the adaptively reduced grid mainly to accelerate the traversal of the volume. As the amount of cells is also reduced in the pre-processing step, memory can be saved. Integration is accelerated by the reduction of sample points. We have also implemented the ray-casting of iso-surfaces in the reduced volume.

The main problems to be solved are the efficient interpolation of values and the gradient. Both problems are solved using a combination of LU-factorization and interpolation described later on. Another problem is the cell traversal, that usually results in a search procedure to find the cell, that contains a certain sample point. Searching for traversing the volume is in our case avoided by combining the sample strategy with the topology of the tetrahedral grid.

## 2 VOLUME RAY CASTING OF IRREGULAR GRIDS WITH IMPLICIT ADAPTIVITY

The algorithm described in this section consists of three main parts:

1. Initialization and preprocessing
2. Ray entrance calculation
3. Ray casting

In the first part of the algorithm the necessary data structures are initialized. Also computations that are necessary for the interpolation and ray intersection calculation with the cell boundaries are performed in advance.

In the ray entrance calculation the volume entrance cell of the ray is computed. The ray tracer, that ray traces the image has been adapted, to make this computation as quick as possible.

Afterwards the volume has to be traversed and the ray has to be integrated. The traversal of the irregular grid brings up some stability problems that will be discussed afterwards.

### 2.1 Initialization and Preprocessing

As an initialization the vertices and tetrahedra are arranged in lists. For every tetrahedron an interpolation function is computed. The interpolation function is of the form

$$f(x, y, z) = a + bx + cy + dz \quad (1)$$

With the four given vertices of the tetrahedron their position and their values the coefficients for the interpolating function

can be easily computed by

$$\begin{pmatrix} 1.0 & x_1 & y_1 & z_1 \\ 1.0 & x_2 & y_2 & z_2 \\ 1.0 & x_3 & y_3 & z_3 \\ 1.0 & x_4 & y_4 & z_4 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} \quad (2)$$

where  $f_i = f(x_i, y_i, z_i)$  are the function values at the tetrahedra vertices and  $(x_i, y_i, z_i)$  are the corresponding coordinates. With the resulting coefficients interpolation can be performed very easily later on. For intersection calculations and the search procedure it is necessary to calculate the normal forms of the triangles, that are faces of the tetrahedra. This is also done in the preprocessing. Afterwards the vertex list is deleted, as the information about the vertices themselves are not used any more. In the end we have an array of tetrahedra, where every tetrahedron contains four triangles. The triangles are enumerated that way, that a relation of the neighboring tetrahedron is implicitly given.

## 2.2 Ray entrance calculation

In this work we visualize medical data sets. These data sets have a cubic definition domain. This property may be used to write a fast volume entrance calculation algorithm, that will be described in the following sections. But especially in combination with finite element methods, more complex definition domains are of interest. In these cases the presented entrance calculation algorithm will not be appropriate. A possible approach to solve this problem is described in the section 5.

To perform the integration of the ray, the first cell the ray intersects with, has to be computed. The volume entrance point is computed by clipping the ray with a cube, as the initial regular grid always fits into a cube. Then, the cell that contains this volume entrance point is searched. This search is performed iteratively. If the actually regarded cell does not contain the point, the regarded cell of the next iteration will be one of the neighboring cells of the actually regarded cell.

To estimate the next cell, the coordinates of the entrance point are inserted in the normal forms of all bordering triangles of the actual tetrahedron. The triangle, that produces the smallest value will be the face triangle of the actual and the next tetrahedron. If the computed value for a triangle is negative, the triangle is separating the entrance point of the ray from all inner parts of the tetrahedron. If there is more than one negative value, the smallest value characterizes the triangle, where the according plane has the highest distance to the entrance point. This is explained in Fig. 2.

To make the searching as easy as possible, the start cell is always the cell that contained the entrance point of the last ray. As the ray tracer of the image is tracing systematically the new entrance point should not be too far away from the old one. To increase coherence the ray tracer scans the scan-lines always in reverse direction than the previous one.

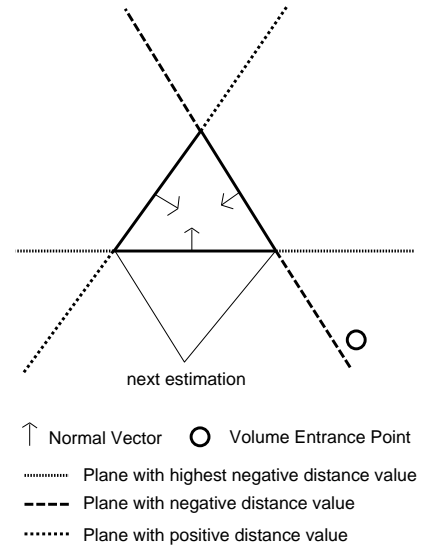


Figure 2: Explanation of the searching scheme

## 2.3 Ray casting

Once the entrance cell of the ray has been found, the ray will be traced in front to back manner. The sample points of the ray are always on the surface of the tetrahedron. This sampling technique is based on an idea of Max et al. [MHC90]. With certain assumptions about the interpolation functions they perform an analytical integration. So the processing of an integration step consists of the following sub-tasks:

1. Compute the exit point of the ray and get the neighboring tetrahedron
2. Compute the length of the ray
3. Interpolation on the faces
4. Perform integration

The exit point is computed using some kind of  $\alpha$ -clipping (see [FvDFH92]). Only the three triangles, that are not the ray entrance triangle of the tetrahedron are taken into considerations. The  $\alpha$ -clipping routine also says which triangle was hit by the ray, what is used to look for the next tetrahedron the ray enters. This is done using the connectivity information. As the ray direction vector is normalized in advance, the length of the ray part within the tetrahedron is given by the generated  $\alpha$ -value. The integration of the ray-part is performed using the trapezoid rule. The location of the sample points is demonstrated in Fig. 3. In this figure one can also clearly see, that in areas of fine triangulation the integration step size gets shorter, what makes the effect of implicit adaptivity. The interpolation of the function value is done using equation (1).

Stability problems during the ray integration may occur if there are two  $\alpha$ -values in the  $\alpha$ -clipping procedure, which

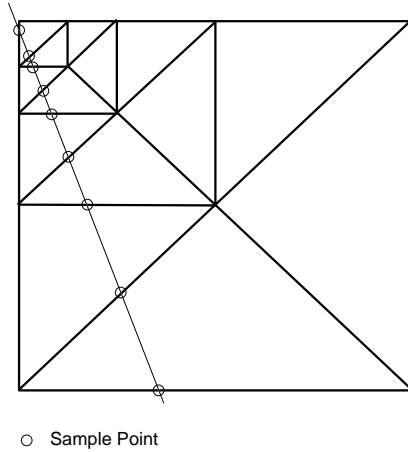


Figure 3: Sample points for a 2D simplicial mesh

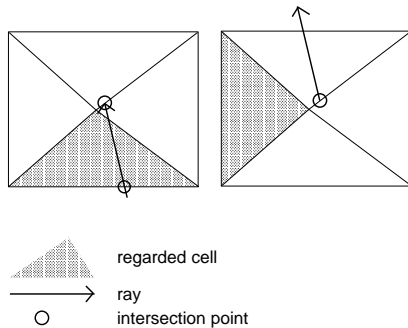


Figure 4: Exception situation in mesh traversal

are nearly equally, or if the length of the integrated ray part within the tetrahedron is very small. In this case the tetrahedron, evaluated to be the next one, might not contain the exit point of the actual tetrahedron. To avoid this case a special correction of the exit point is made, if the integration length is very small or if there are two  $\alpha$ -values, which are pretty much the same. In this case the computed exit point is moved a little bit towards the mid point of the next tetrahedron to be traversed. What would happen if this exception would have not been handled is shown in Fig. 4.

### 3 RAY CASTING OF ISO SURFACES ON IRREGULAR GRIDS

To extend the volume ray casting, iso-surfaces have been implemented. As the interpolation function (1) is affine one can easily check, if an iso-surface is hit during the traversal of the tetrahedron, if the interval, spanned by the value at the entrance point and the value at the exit point, contains the iso-value. The exact intersection point can be computed

using linear interpolation due to the linear interpolation function. As a shading model we use the phong model with pure diffuse reflection. The transparency of the surface can be defined but is constant.

To perform the shading algorithm the normal of the surface is needed. As in most iso-surface algorithms we use the gradient to compute the normal. The differentiation of equation (1) results in the gradient

$$\nabla f = \begin{pmatrix} b \\ c \\ d \end{pmatrix}. \quad (3)$$

The application of this formula results in surfaces that look flat shaded like, as the resulting gradient is not continuous over the whole volume.

To avoid this effect the shading normal is computed differently in our case. First equation (3) is used to compute gradients for every tetrahedron. Every tetrahedron adds its gradient to its four vertices, which initially have a zero vector as gradient. This way every vertex contains a volume weighted normal vector of the tetrahedra it belongs to. In a third iteration a set of interpolation functions for the interpolation of the three gradient components is computed. This is done by solving the equation (2) for all three gradient components. This is done in advance with the preprocessing. To perform the shading, the gradient is interpolated at the estimated surface intersection position.

## 4 EXPERIMENTS

To analyze the characteristics of the presented algorithm several experiments have been performed. Our main aim was to analyze the relation of computation time and image quality. For this purpose a chromosome data set has been used. This data set originates from the institute of anatomy, University of Giessen. It consists of  $256^3$  voxels.

The chromosome data set, ray-traced the traditional way with two samples per voxel, is shown in image 6f. All images are ray-traced at a  $500 * 500$  pixels per image resolution. The other images 6a - 6e show the same data set with the same transfer functions for different reduction stages. The amount of vertices and tetrahedra for every data set left, is displayed besides the used computational time in table 4. For the original data set in plate 6f the amount of cubic cells have been inserted.

The image differences between the original image in plate 6f and the reduced image with most tetrahedra left, that is displayed in plate 6e is marginal. The number of vertices has been reduced to 0.2%. The image ray-tracing time has been reduced down to 20%. These numbers show the strength of the combination of the finite element analysis algorithm with the inherent adaptive ray-casting method. The analysis algorithm tends to finer tetrahedrization of regions with high gradient magnitude. This can be clearly seen at the slope of

plate	cells	vertices	time [sec]
6a	12,570	2,320	61
6b	25,571	4,613	74
6c	50,817	9,076	93
6d	111,891	19,769	117
6e	194,885	34,269	148
6f	16,581,375	16,777,216	727

Table 1: Computational time used for different resolutions

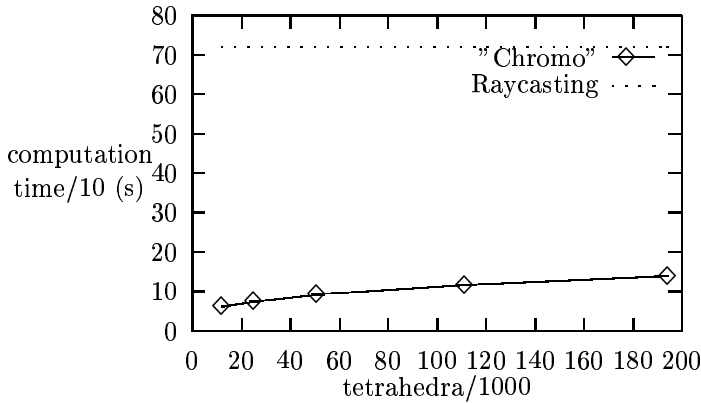


Figure 5: Times for the ray-tracing of the chromosome data-set

the chromosome in the images 6a - 6f. This slope degenerates at the lower levels of resolution very quickly. On the other hand the black kernel, which is the nucleus in the upper left corner of the chromosome remains quite stable even in levels of lower resolution.

This has the consequence, that the main features of a volume are still intact at lower resolutions, what makes this technique interesting for interactive preview, as these drastically reduced representations are still useful for deciding, whether there are interesting features in the data-set or not. In plate 6a for instance, the amount of vertices has been reduced down to 0.01% of the original data set, and the required computational time has been reduced down to 8.4% of the original data set. The main characteristics as the orientation of the chromosome and the position of the nucleus are still visible.

The computation time for the original data set and for the reduced ones has been visualized in image 5. The dotted line represents the computation time of the original data-set with standard volume ray-casting technique. The computation time for the reduced data sets with respect to the number of tetrahedra left, is shown in the scatter plot. One can see, that the function is nearly linear except for the first samples. The reason for this phenomenon is, that the ray-entrance calculation consumes relatively more time, if there are just few tetrahedra, that are traversed along a single ray.

The presented volume ray-casting acceleration technique is orthogonal to the most known volume ray-tracing acceleration techniques. In other words the combination of the presented acceleration technique with one of the known ones results in extra acceleration. As an example early ray-termination has been implemented and experiments with the chromosome data set have been made. The images shown in plate 6f and 6e have been ray-traced using a terminating opacity of 0.8. The time needed for the original data set was 707 seconds and for the reduced data set with 34269 vertices left, the computational time was 138 seconds. The acceleration factor of five remained nearly constant.

Further the ray-tracing of iso-surfaces has been implemented. As an example we have chosen a MRI-scan of a head. The original data set contains  $256 * 256 * 113$  voxels. In plate 6a the volume rendering of the reduced data set with 70372 vertices left is shown. In plate 6g an iso-surface of this reduced data set is shown. Due to the interpolation of the weighted gradients, a Gouraud shading effect is produced.

## 5 CONCLUSIONS AND FUTURE WORK

We have shown that implicit adaptive volume-ray-casting is a powerful method for volume-ray-casting acceleration. The basic idea to generate the adaptivity information in a preprocessing step, and to code it implicitly into a data structure offers the possibility to construct special visualization algorithms, that make use of this implicit adaptivity information.

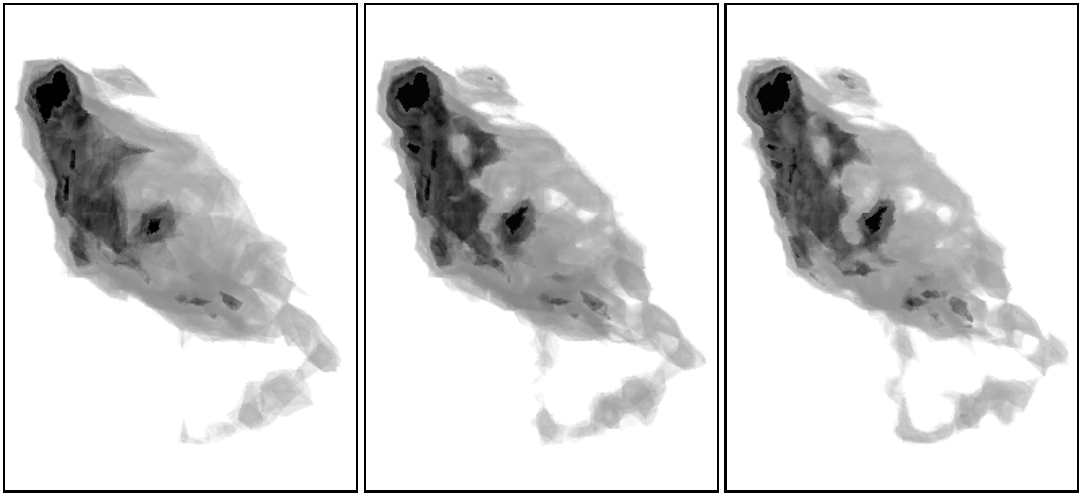
We are currently planning to extend this approach to volumetric defined complex domains. The problem in this case is to find the ray-entrance point of the volume. Another problem is the fact, that there may be multiple ray-entrance and exit points, if the domain is not convex. An efficient solution to this problem might be to extract the boundary triangles of this volume first. The ray-entrance and exit points may be computed by using a scan-line orientated algorithm. The scan-conversion would be performed using a modified z-buffer algorithm, that stores every fragment with its z-values, that are sorted after the scan-line algorithm has been terminated. This way one gets every volumetric entrance and exit point along the ray. The traversal within the definition domain would again be done using the connectivity information.

## References

- [Ban96] R. E. Bank. Hierarchical Bases and the Finite Element Method. *Acta Numerica*, 1996.
- [CDFM<sup>+</sup>94] P. Cignoni, L. De Floriani, C. Montani, E. Puppo, and R. Scopigno. Multiresolution Modeling and Visualization of Volume Data

- based on Simplicial Complexes. In *Proceedings 1994 Symposium on Volume Visualization*, pages 19–26, 1994.
- [DH92] J. Danskin and P. Hanrahan. Fast algorithms for volume ray tracing. In *Transactions 1992 workshop on Volume Visualization*, pages 91–98, 1992.
- [FT86] A. Fujimoto and K. Tanaka, T. Iwata. Arts: Accelerated ray-tracing system. In *IEEE CG&A*, pages 16–26, April 1986.
- [FvDFH92] Foley, van Dam, Feiner, and Hughes. *Computer Graphics Principles and Practice*, volume 2. Addison-Wesley, 1992.
- [Gar90] M.P. Garrity. Raytracing irregular volume data. *ACM Computer Graphics*, 24(5), 1990.
- [GGS95] M. H. Gross, R. Gatti, and O. Stadt. Fast Multiresolution for Surface Meshing. In *Proceedings IEEE Visualization '95*, pages 135–142, October 1995.
- [GLE96] R. Grosso, C. Luerig, and T. Ertl. Adaptive multilevel finite elements for mesh optimization and visualization. Technical report, Computer Graphics Group, University of Erlangen-Nuernberg, 1996.
- [LE96] Ch. Luerig and Th. Ertl. Adaptive Iso-Surface Generation. In B. Girod, H. Niemann, and H.-P. Seidel, editors, *3D Image Analysis and Synthesis '96*, pages 183–190, 1996.
- [Lev90] M. Levoy. Display of surface from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, 1990.
- [MHC90] N. Max, P. Hanrahn, and R. Crawfis. Area and Volume Coherence for Efficient Visualization of 3D Scalar functions. *Computer Graphics*, 24(5):27–33, November 1990.
- [Wes95] R. Westermann. Compression domain rendering of time-resolved volume data. In *IEEE Visualization 1995 Conference Proceedings*, pages 51–58, 1995.
- [WVGTG96] J. Wilhelms, A. Van Gelder, P. Tarantino, and J. Gibbs. Hierarchical and parallelizable direct volume rendering for irregular and multiple grids. In *Proceedings Visualization '96*, pages 57–64, 1996.

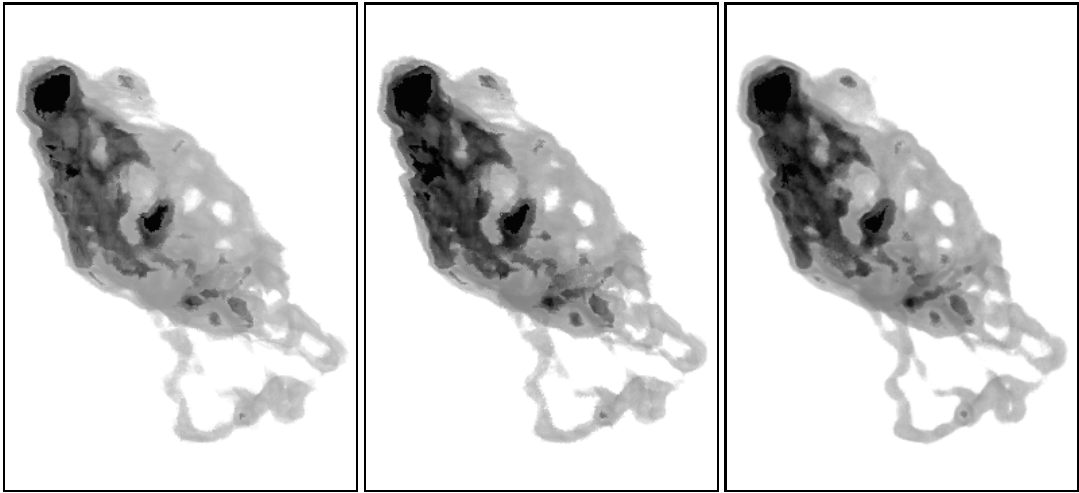
Lehrstuhl für Graphische Datenverarbeitung (IMMD9)  
 Universität Erlangen-Nürnberg  
 Am Weichselgarten 9  
 91058 Erlangen  
 Germany  
 Email:  
 cpluerig@immd9.informatik.uni-erlangen.de



(a) 0.01% of the vertices

(b) 0.03% of the vertices

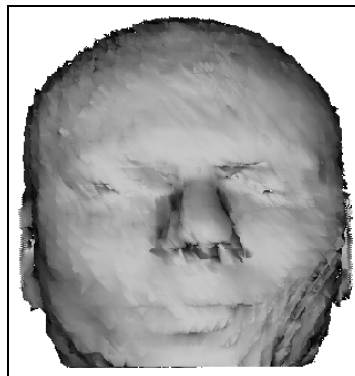
(c) 0.05% of the vertices



(d) 0.1% of the vertices

(e) 0.2% of the vertices

(f) full resolution



(g) iso surface 1% of the vertices

Figure 6: Image plate showing the results of volume ray-casting