

# Adaptive Iso-surface Generation

Christoph Lürig and Thomas Ertl

University of Erlangen, IMMD IX, Computer Graphics Group  
Am Weichselgarten 9, D-91058 Erlangen, Germany  
Email: {cpluerig,ertl}@informatik.uni-erlangen.de

## Abstract

In this work a method for adaptive iso-surface generation on reduced data sets is presented. The aim is to accelerate the generation and rendering of iso-surfaces. The amount of cells in areas of low spatial frequency is reduced. In this way less cells have to be analyzed and less surface triangles are generated in the mentioned regions. First, the vertices of the original uniform grid are reduced. This is done either by an edge detection algorithm or by a maximum error constrained reduction and reconstruction algorithm. The resulting vertices are tetrahedrized with a Delaunay triangulation algorithm. The iso-surfaces are extracted by a sweeping simplicies algorithm is used that pre-selects the relevant tetrahedrons for a given iso value.

## 1 Introduction

Iso-surfaces are very often used for the visualization of medical scalar fields. The generation of iso-surfaces is an indirect visualization method, that is quite expensive especially for huge medical data sets. The aim of this work is to develop an accelerated iso-surface generator for scalar volume data on an orthogonal equidistant grid. To reach this aim we follow two basic ideas. The first idea is that the orthogonal equidistant grid is transformed into an irregular grid, which has less grid points than the original one. For this purpose the original data set is down-sampled and the resulting vertices are tetrahedrized by a 3D Delaunay tetrahedrization algorithm. The second idea is to extract iso-surfaces using a special marching cubes algorithm. This algorithm sorts the tetrahedrons by their value range to avoid checking unnecessary tetrahedrons during iso-surface generation.

In this way the generation of iso-surfaces is accelerated to some extent and interactive surface browsing and the navigation in the results is possible for even lower powered machines.

The presented approach differs from the one described in [7], in that they are using the tetrahedrization and a vertex selection algorithm iteratively. The approach presented here uses two independent phases.

The different processing stages are visualized using an artificial data set. It is represented in a  $50^3$  volume of shorts. The values in a sphere around the center are at constant value and are decreasing towards the borders proportional to the inverse distance to the center of the cube.

## 2 Iso-surface Generation

### 2.1 Volume sampling

The first step in an adaptive iso-surface generation technique is to reduce the generated vertices. The criterion for the decision which vertex to keep provides the adaptivity aspect. Generally saying, the higher the space frequency in a region is the more vertices should be kept. In [4] this approach is accomplished by the application of the 2D wavelet transformation for a two dimensional analysis. We use two different approaches that may be applied alternatively or in combination. The first one uses the local maxima of the gradients, which may also be evaluated by the Canny operator, which is a special case of the continuous wavelet transformation. The second approach uses some kind of octree analysis. This method has the advantage that a maximum error for a reconstruction by trilinear interpolation may be set. This can not be done by wavelet analysis. The combination of the local maximum method and a reconstruction criterion is suggested in [3]. The two re-sampling criterions

are discussed in the next paragraphs.

**Structural Analysis:** The aim of the structural analysis is to extract the local maxima of the gradient. In the implemented tool the computation of the gradient can either be done by the 3D extension of the canny operator or by the simple finite difference quotient. In most cases the quotient seems to give sufficient results.

In the structural analysis a vertex is kept if the magnitude of the gradient exceeds a user specified threshold and is locally maximum. The magnitude of the gradient is a local maximum if the values at the two neighboring voxels, that lie in line with the direction of the gradient, are lower than the value of the regarded voxel. For at least one neighboring voxel the value must be strictly lower. To choose the neighboring voxels rounded inclination angles of the gradient are used. This technique is described in [6].

The three dimensional canny operator computes the gradient by convoluting the volume data with the partial derivatives of a three dimensional Gaussian. The variance of the Gaussian and the size of the convolution kernel can be specified in the implemented tool. The convolutions for each of the partial derivations are evaluated separately. In effect the gradient of the lowpass filtered volume is computed. The generated vertices of the demonstration data set are shown in Figure 1. In this case the canny operator and the finite difference quotient return the same result.

**Reconstruction analysis:** The reconstruction analysis is an octree based analysis of the dataset. The principal idea is that all vertices of a sub-cube, but not the corner vertices of this cube, may be eliminated, if it is possible to reconstruct all inner vertices of this cube using trilinear interpolation without introducing an error which is higher than a specified bound. For this purpose the maximum norm is used. The idea of the octree is only used implicitly but an octree is never constructed explicitly. The idea of an octree – respectively quadtree – based analysis is also used in region growing [1].

To perform the analysis the volume is conceptually extended to a cubic size that is a power of two. This is necessary to make a successive octree based region splitting possible. The main anal-

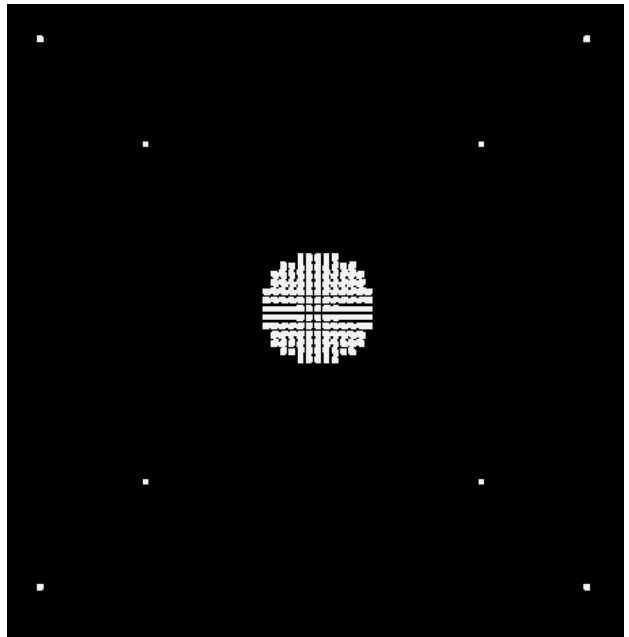


Figure 1: Local maxima of the demonstration data set with corner vertices of the volume

ysis algorithm works recursively. It starts with the sub-cube that contains the whole extended volume.

If a cube is processed all sub-cubes are analyzed if the regarded cube is not atomic. A cube is atomic if it consists of  $3^3$  voxels. If all eight sub-cubes return success in their analysis, then further steps on that cube are performed. First, the error for the vertices shown in Figure 2 are estimated. These errors are propagated down to all sub-cubes to estimate the error increment caused if the regarded vertices would be deleted. The maximum of all these errors is evaluated. If the maximum is lower than a specified threshold, then the regarded vertices are marked for deletion. Not all vertices may be deleted directly because they may be corner edge or side vertices of other sub-cubes. These cases are handled separately using counter variables. If the resulting error is too high, then the added error increments are erased. The handling of the error increments has to be performed carefully, as vertices may be shared by different cubes. If the error does not exceed a certain bound the actual cube has been processed successfully.

To estimate the local error the trilinear interpolation does not have to be performed explicitly. This is due to the fact that the analyzed vertices are always half way between the corner vertices of the analyzed cube. The inter-

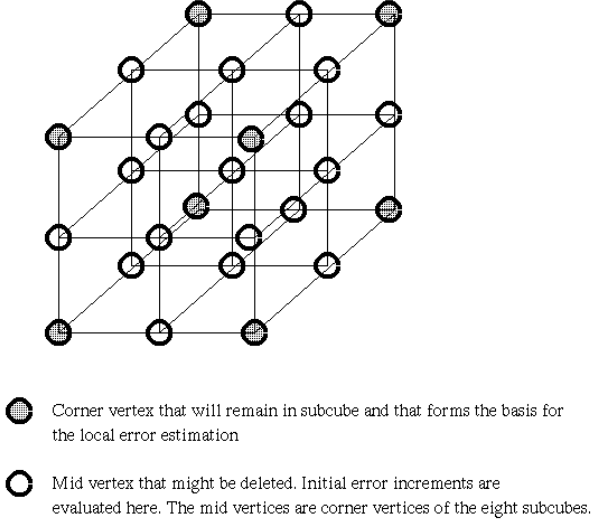


Figure 2: Vertices analysis scheme of a sub-cube

polated value of an edge midpoint is the average of the two neighboring corner vertices. The interpolated value of a side midpoint is the average value of the four vertices of this side. The value of the cube midpoint is the average value of all eight vertices belonging to this cube.

If the cube is not atomic, then the errors of the other vertices have to be updated. This is done incrementally. The interpolation errors on the mid vertices are computed. The error increment is propagated to all eight sub-cubes. For all sub-cubes the new error increments are computed the same way as it was discussed in the previous paragraph. The evaluated error increments are added to the actual errors. Special care has to be taken, that an error increment is not added twice or more if the regarded vertex belongs to more than one sub-cube of the same level. This is the case if it is an inner vertex. If the evaluated error was too high, the error increment is then reverted by propagating the negative error increment.

The same problem occurs during the elimination of vertices. A vertex should only be eliminated if all sub-cubes the vertex belongs to, allow the elimination. To handle this problem a counter for each vertex is installed. Every time a sub-cube allows the elimination of a vertex it increments its counter. Depending on the type of the vertex it is eliminated if its counter reaches a certain limit. For the case that the considered vertex does not lie at the border of the volume the limits are shown in Table 1. If this is not the

vertex position	necessary counts
edge mid-point	4
side mid-point	2
cube mid-point	1

Table 1: Necessary counters for the vertex elimination in the standard case

vertex position	necessary counts
edge mid-point	2
side mid-point	1
cube mid-point	1

Table 2: Necessary counters for the vertex elimination if the vertex lies at a side of the volume

case the limits are lower. One has to distinguish whether the vertices lie on an edge (Table 3) of the volume or just on a side (Table 2).

The artificial sphere volume data set has been analyzed with a maximum error of 0.05. The resulting vertices are shown in Figure 3. The phenomenon that the vertices are positioned asymmetrically in the symmetric data set is caused by the volume extension before the analysis starts.

## 2.2 Delaunay Triangulation

The resampling of the volume reduces the amount of vertices but the original cell structure of the orthogonal equidistant grid is lost. Iso-surface algorithms require a unique cell structure as single cells are analyzed for triangle output. To get consistent iso-surfaces there should not be any cranks in the generated cell structure. Due to the structural analysis nothing can be said about the position of the vertices that pass the filter process. To fulfill the mentioned requirements we decided to generate an irregular grid for the

vertex position	necessary counts
edge mid-point	1
side mid-point	1
cube mid-point	1

Table 3: Necessary counters for the vertex elimination if the vertex lies at an edge of the volume

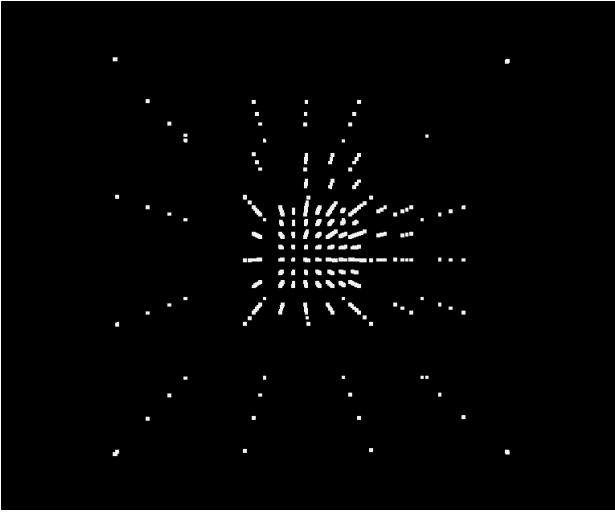


Figure 3: Vertices of the reconstruction analysis of the demonstration data set at an error level of 0.05

vertices. This grid consists of tetrahedrons.

The grid generation method that is most often used for the generation of irregular grids is the Delaunay triangulation. A triangulation is called Delaunay if all of its cells are local Delaunay. In two dimensions a triangle is local Delaunay if no other vertex lies within the smallest circle that encloses the triangle. In three dimensions a tetrahedron is said to be local Delaunay if no other vertex lies within the smallest sphere that encloses the tetrahedron. The practical consequence of this method is that degenerated tetrahedrons can only occur at the border of the data set. Degenerated tetrahedrons should be avoided as they can result in degenerated surface elements in the iso-surface.

The Delaunay triangulator that has been used in this work is described in [2]. It successively adds all vertices to the triangulated volume. The new vertices must always lie outside the convex hull of the triangulated area. If a new vertex is added, tetrahedrons to all surface triangles are constructed, if they do not intersect with the rest of the triangulated area. If a local Delaunay property is destroyed, then the old tetrahedron is deleted, and the three remaining surfaces are added for triangulation. In the worst case all those vertices which have been triangulated so far must be triangulated again. As a consequence, the triangulation algorithm has a time complexity of  $O(n^2)$ . This process is shown in Figure 4 for the two-dimensional case. Especially for three

dimensions there are many exceptions which are discussed in [2] in more detail.

Most of the Delaunay triangulation algorithms as this one assume, that no four points of the triangulated volume are coplanar. This restriction is also mentioned in [9]. In the presented work the vertices always have integer coordinates and consequently this case occurs quite often. In [9] they propose to conceptually perturb the position of the vertices. In this work another approach is followed as perturbation would mean to switch from integer coordinates to float coordinates. If there are at least four coplanar vertices it might happen that a degenerated tetrahedron is constructed. For this tetrahedron it is impossible to check the local Delaunay property. If such a tetrahedron is about to be constructed, the tetrahedron that provides the base triangle for the construction is deleted. The left surface triangles are added to the triangulation. In this way the construction of degenerated tetrahedrons is avoided.

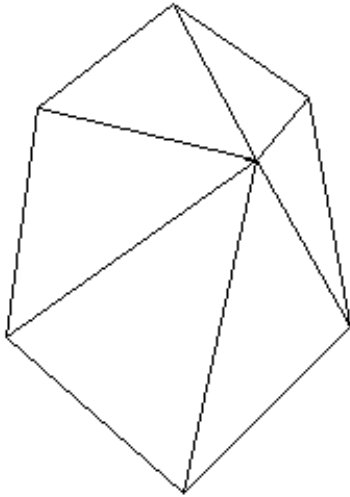
Two images are presented to show the tetraedrization results. In Figure 5 the vertices of the structural analysis of the demonstration data set and its corner vertices are tetraedrized. In Figure 6 the same is done for the reconstruction analysis.

### 2.3 Computation of Gradient

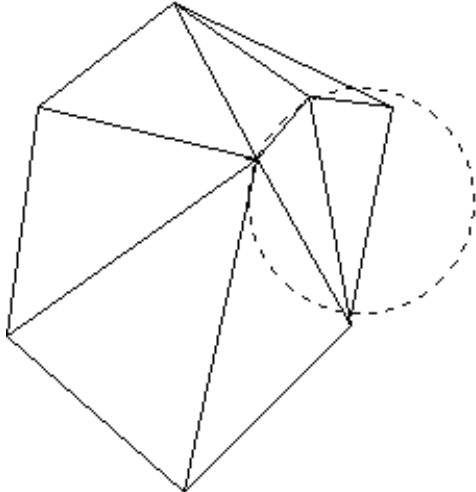
The computation of the gradient at the position of the vertices is important to compute the shading normals for the iso-surface rendering. The simple computation of the gradient using the finite difference method turned out to be not sufficient. The resulting shading normals were not consistent with the geometry. The numerical methods to compute the gradient use only local information around the vertex. There may be changes in the values around this vertex that are of such a small size, that they do not affect the next vertex in the triangulated area. On the other hand there might be a change in value between two vertices belonging to the same tetrahedron. But this change in value does not happen in the neighborhood of either of the vertices. Thus, it is not reflected in the resulting gradient.

To solve this problem the gradient has to be computed with the information of the vertices used for triangulation. In this work this is done by computing the gradient of an approximation

Area triangulated so far



New vertex added new triangles built and local Delaunay property destroyed



Defect triangle is retetrahedrized

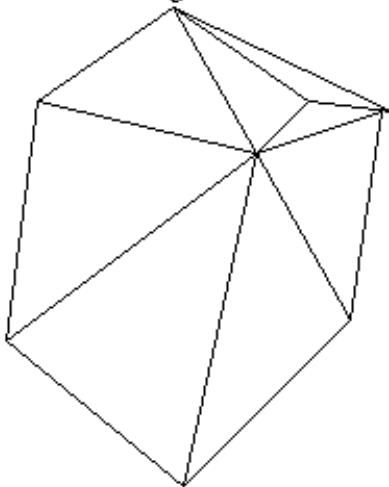


Figure 4: The Delaunay triangulator

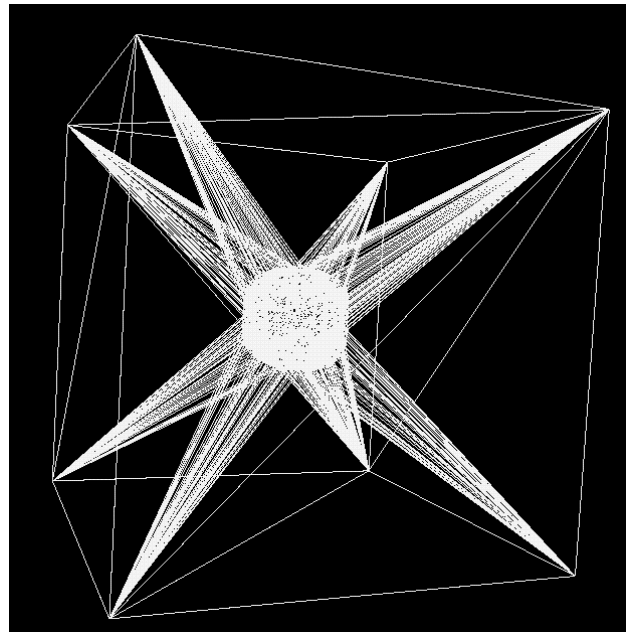


Figure 5: The triangulation of the structural analysis output

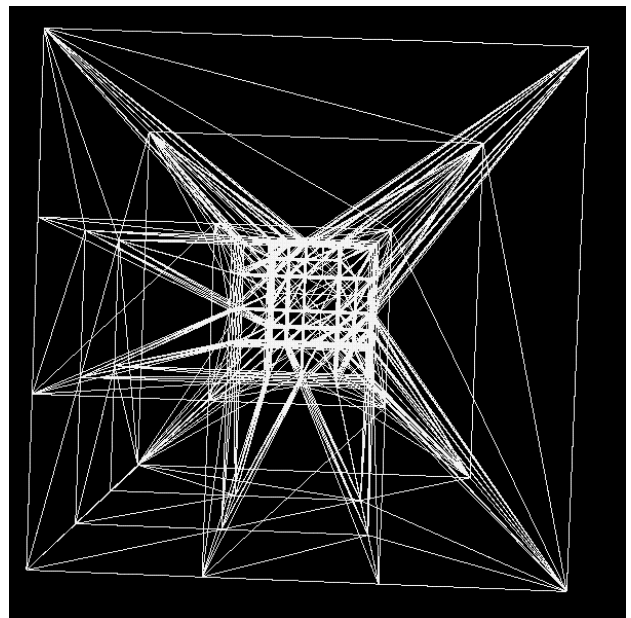


Figure 6: The triangulation of the reconstruction analysis output

function

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

For the approximation the regarded vertex itself and all vertices that are connected with an edge to the regarded vertex are taken into account. The coefficients are computed using a linear least squares approximation. The approximation is computed with a singularity value decomposition:

$$\nabla f \approx \nabla g = \begin{pmatrix} b \\ c \\ d \end{pmatrix} . \quad (2)$$

The resulting gradient is consistent with the geometry.

## 2.4 Sweeping Simplices

When the generation of the unstructured grid and the computation of the gradient is completed, the iso-surface itself is generated. To solve this task the approach proposed in [11] has been followed. The basic idea is to make a selection of the tetrahedrons by their value range and to generate triangles from the selected tetrahedrons as it is done in the marching cubes algorithm.

At first the tetrahedrons are sorted in a binary tree. Every node in the binary tree may contain several tetrahedrons. The depth of the tree is specified in advance. Every node belongs to a value interval, the two sub-nodes of a node split the intervals in half. A tetrahedron belongs to the node with the smallest interval that covers the whole value interval of the tetrahedron. The split points of the intervals are chosen in a way that the tree becomes as balanced as possible. If an iso-value is set, the leaf node with the interval that contains the iso-value is searched. From this node starting the tree is traversed to the root. Only those tetrahedrons are analyzed that lie along the traversed path.

Every node is annotated with two lists. The first one is a list of the tetrahedrons sorted by their minimum values. The second list is sorted by the tetrahedrons maximum values. Every entry in this second list also contains a flag. Every entry in the first list contains a pointer to an entry in the second list. The positions of the old iso-value are marked in both lists. When the iso-value is changed, then the first list is traversed from the position of the old iso-value to the new

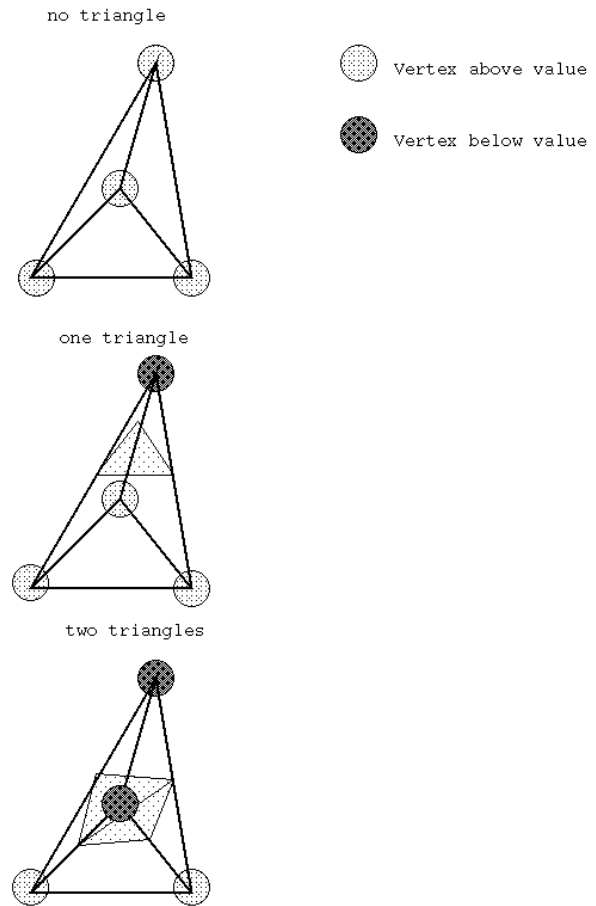


Figure 7: Principle possibilities for triangle output

one. All pointers are followed to the second list and the corresponding flags of the entries are updated. Afterwards the second list is traversed from the position of the iso-value to its end. All tetrahedrons that have their flags set are analyzed. In this way a small change in the iso-value requires only little computation.

The vertices of each selected triangle are analyzed whether their value is above or below the selected iso-value. Depending on the constellation zero, one or two triangles are generated (see Figure 7). The position of the vertices of the triangles are stored in a look up table. This table contains the edges of the tetrahedron on which the vertices are positioned. The position on the edge is estimated by linear interpolation. The gradient is also interpolated lineary to compute the shading normal for the vertices of the triangle. Finally the triangles are ready for Gouraud shading.

The construction of the tree and the lists, that

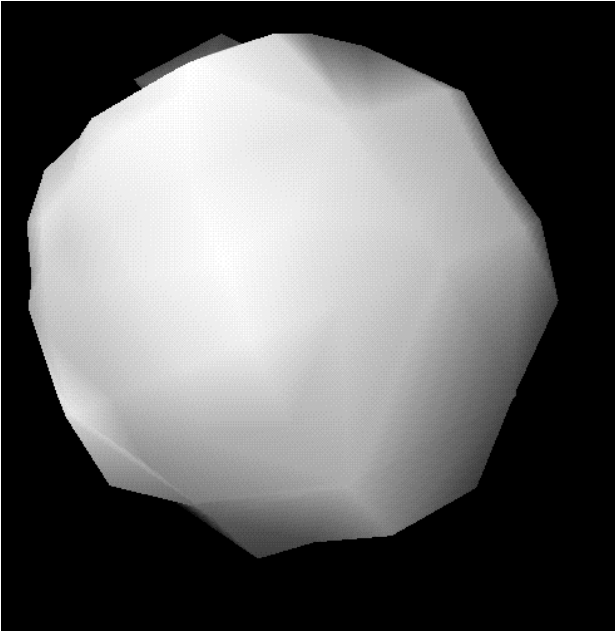


Figure 8: Iso-surface of reduced demonstration data set

are annotating the nodes, is done in advance. For more details see [11].

### 3 Results and future work

In the following, the results of reduction experiments are shown. First, the analytical data set is presented. It has been undergoing a pure reconstructive analysis with a maximum error bound of 0.15. 198 vertices were generated. This leads to a compression ratio of 99.8%. The iso-surface is computed for a value of 0.7. 542 triangles have been extracted in 0.03 seconds. The result is shown in Figure 8. The correct solution would be a sphere.

The following images show the result of a reduction experiment of a medical data set. Using the original dataset with a traditional marching cubes algorithm, 323,822 triangles have been extracted from a  $128^3$  vertices data set in 25.28 seconds. The result is shown in Figure 9.

Figure 10 shows a reduced data set with 18,456 vertices left. A compression ratio of 99.12% is achieved. For the given iso-value of 0.3, 71,872 triangles have been extracted in 2.6 seconds. If the algorithm is used for interactive preview of iso-surfaces, it still generates images with an acceptable quality.

Interesting is the fact that the amount of gen-

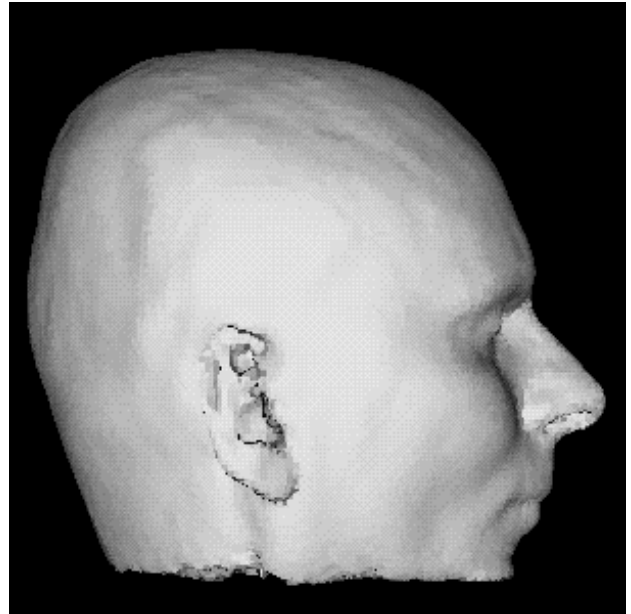


Figure 9: Iso-surface generated with standard marching cube

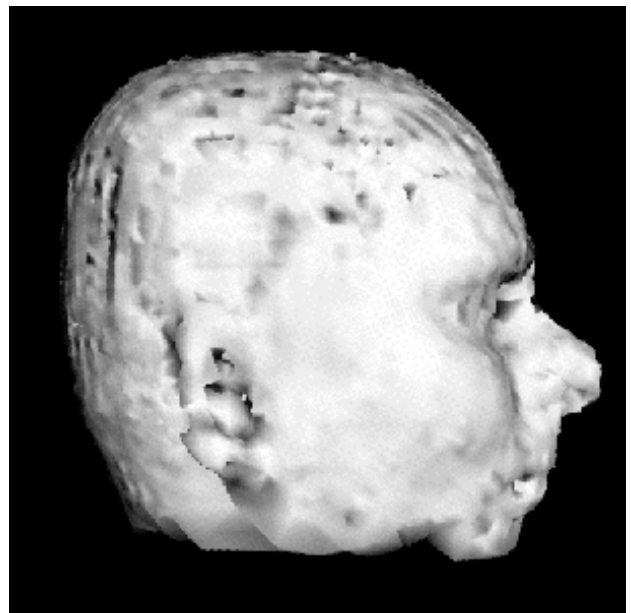


Figure 10: Iso-surface generated with reduced data set

erated triangles is not that much reduced as the amount of vertices. The reason of this phenomenon is that the selected iso-value generates an iso-surface that lies in an important part of the data set. The spatial frequency is very high in this area. Due to the adaptivity property of this algorithm a higher resolution of tetrahedrons is generated in these parts of the volume.

In the future we investigate the computation of  $\alpha$ -shapes [10] of the tetrahedrized volume. Region growing experiments will be performed with the tetrahedrons. This is consistent with the sampling step that is performed before. The sampling and function approximation can be interpreted as a kind of region growing [8].

## References

- [1] Chin-Hwa Lee "Recursive Region Splitting at Hierarchical Scope Views". Computer Vision Graphics and Image Processing 33, pages 237-258 (1986)
- [2] Barry Joe "Construction of three dimensional Delaunay triangulations using local transformations". Computer Aided Geometric Design 8 (1991), page 123-142
- [3] Robert J. Fowler, James J. Little "Automatic Extraction of Irregular Network Digital Terrain Models". Computer Graphics 13(3): pages 199-207, 1979
- [4] M.H. Gross, R. Gatti, O. Staadt "Fast Multiresolution Surface Meshing". Proceedings Visualization 1995
- [5] Leonidas J. Guibas, Jorge Stolfi "Primitives for the manipulation of general subdivision and the computation of Voronoi diagrams" ACM Transactions on Graphics 4 (2) pp. 74-123, 1985
- [6] Mallat, S. and Zhong. S. "Characterization of Signals from Multiscale Edges". IEEE Transactions on pattern analysis and machine intelligence. Vol. 14. No. 7: Juli 1992. 710-732
- [7] Paolo Cignoni, Leila De Floriani, Claudio Montani, Enrico Puppo, Roberto Scopigno "Multiresolution Modeling and Visualization of Volume Data based on Simplicial Complexes"
- [8] Steven W. Zucker "SURVEY Region Growing: Childhood and Adolescence". Computer Graphics and Image Processing 5, 382-399 (1976)
- [9] H. Edelsbrunner, F. P. Preparata, D.B. West "Tetrahedrizing Point Sets in Three Dimensions". Journal of Symbolic Computation (1990),10,335-347
- [10] H. Edelsbrunner, E. P. Mücke "Three-Dimensional Alpha Shapes". ACM Transactions on Graphics, Vol. 13, No. 1, January 1994, 43-72
- [11] Han-Wei Shen Christopher R. Johnsen "Sweeping Simplicies: A fast iso-surface extraction algorithm for unstructured grids", Proceedings of Visualization '95 page 143-150, IEEE Computer Society Press