Simplification of Closed Triangulated Surfaces Using Simulated Annealing

Peer-Timo Bremer, Bernd Hamann,
Oliver Kreylos and Franz-Erich Wolter

Abstract. We describe a method to approximate a closed surface triangulation using simulated annealing. Our approach guarantees that all vertices and triangles in an approximating surface triangulation are within a user-defined distance of the original surface triangulation. We introduce the idea of atomic envelopes to guarantee error bounds that are independent of the surface geometry. Atomic envelopes also allow approximation distance to be different for different parts of the surface. We start with the original triangulation and perturb it randomly and improve an approximating triangulation by locally changing the triangulation, using a simulated annealing algorithm. Our algorithm is not restricted to using only original vertices; the algorithm considers every point inside the envelope triangulation as a possible position. The algorithm attempts to minimize the total number of vertices needed to approximate the original surface triangulation within the prescribed error bound.

§1. Introduction

Over the past two decades, data visualization has become increasingly important in several research areas, including medical, fluid flow and geographical data. The speed of visualization algorithms has unfortunately not kept up with the speed of developing new technology producing high-resolution data. Every year, the quality of imaging and computational simulation technology — including laser scanners, digital cameras and radar systems — improves substantially. This results in such an increase in the amount of data that even state-of-the-art computers are stretched beyond their capacities. However, it has become apparent that for many applications, large parts of data sets are often not necessary for generating a good picture. The goal was and still is to reduce data sets in such a way that the pictures generated from a reduced data set are highly similar to those produced from the original one.

We are concerned with polygonal surfaces and their compression. Examples for polygonal surfaces are discretized height fields, parametric surfaces,

Mathematical Methods in CAGD: Oslo 2000
Tom Lyche and Larry L. Schumaker (eds.), pp. 1–8.

ISBN 0-8265-xxxx-x.

and manifold surfaces. We focus on triangulated two-dimensional (2D) manifolds with no boundaries. For an extensive overview of the field of polygonal surface simplification, we refer to Heckbert and Garland [6] and Rossignac [14].

We present a randomized algorithm that approximates triangulated, orientable 2D manifolds without boundaries, using a *min-#* approach, see §2. The algorithm preserves a specified error bound.

§2. Related Work

2.1 Two Approximation Types: Min-ϵ and Min-#

When approximating a polygonal surface using the min-ϵ approach, one has to determine, for a given number n, an approximation that consists of n vertices and minimizes the approximation error. Many of the common algorithms use min-ϵ optimization, and several references are given in [6, 14]. Of special interest is Kreylos and Hamann [9], since they use a method closely related to the one presented here.

Using a min-# approximation approach, one tries to find an approximation with the minimal number of vertices that satisfies a tolerance condition [2]. This approach is relevant for scientific applications. For example, given the size of an object and the view-point distance, one can compute the error tolerance related to one pixel on the screen. Approximating the object within this tolerance results in a picture where each data point is no more than one pixel away from its original location. Computing min-# approximations can be very complicated and expensive. The error metric one wants to minimize is the number of vertices, faces or edges. Additionally, one has to stay inside an error bound. Our algorithm ensures that no point of the approximating surface deviates more than ϵ from the original surface. It is important to notice that this condition is stronger than to require that only the vertices be inside an error bound. It requires us to consider an *offset* around the original surface, and the approximation surface must stay inside this offset. Such an approach was first proposed by Cohen et al. [2], and was called simplification envelope. A simplification envelope is a linearized and, in some respects, simplified version of the exact offset.

2.2 Simplification Envelopes

The simplification envelope of a triangulated surface is constructed in the following way: For each vertex, one computes its normal $\vec{n}$ as a combination of the normals of the surrounding triangles, normalized to length ϵ; one defines two offset vertices, the $(+\epsilon)$-offset and the $(-\epsilon)$-offset vertices, by adding/subtracting $\vec{n}$ to/from the original vertex. This defines a so-called fundamental prism.

This approximation of the offset is close to the exact one as long as the original surface has low curvature. Our approach uses this type of envelope, but it provides the option to use better approximations.

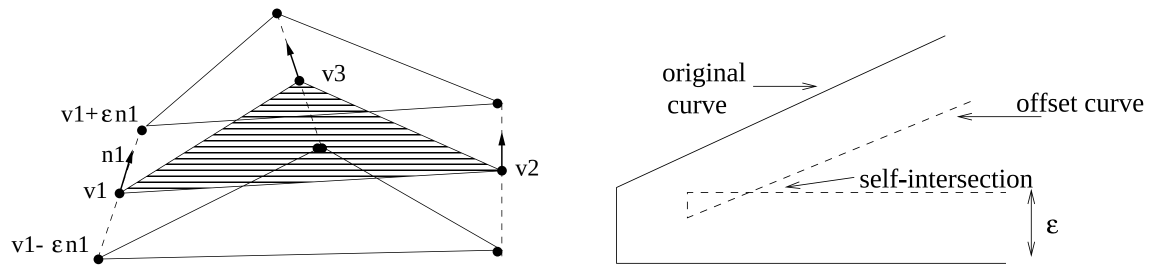

Fig. 1. (a) Fundamental prism; (b) self-intersecting offset curves.

The second problem is caused by self-intersections, see Fig. 1b. Cohen et al. [2] require a simplification envelope that does not self-intersect. They use the global ϵ-value whenever possible and decrease it in areas of possible self-intersections. Our approach is not impacted by self-intersections, and can handle every ϵ-value at any given vertex.

§3. Atomic Envelopes

To satisfy an a priori error bound, we define atomic envelopes. For each triangle, we construct an atomic envelope so that the simplification envelope equals the union of atomic envelopes. Our implementation uses fundamental prisms as atomic envelopes but different constructions are possible when higher accuracy is desired.

During simplification, we have to decide whether a triangle lies inside the simplification envelope. To answer this query we first find all atomic envelopes that might intersect the triangle. We are only interested in the top and bottom triangles that intersect the triangle in question. We use a bounding box test incorporating an R*-Tree [8] to speed up calculations. Especially for smaller error bounds, this results in roughly the same set of triangles one would get using triangulated offset surfaces to describe the simplification envelope. We intersect all resulting triangles with the triangle being tested. At each resulting intersection point, the triangle might leave the simplification envelope. It leaves the envelope if and only if the exit point is not covered by another atomic envelope. To test this, we use the fact that fundamental prisms are pentaeder Bézier volumina [10, 11] and solve the resulting non-linear system of equations.

Cohen et al. [2] define the side faces of a fundamental prism as bilinear patches, defined by the four corner points. Since we deal with closed triangulated surfaces, we do not have to consider the side patches. A triangle cannot leave the envelope through a side patch of a prism: The fundamental prisms of two neighboring triangles always share a common side patch, since the side patches correspond to the edges of the triangulated surface; thus a triangle leaving a prism through a side patch immediately enters another prism.

§4. Simplification

We simplify the given surface using a *simulated annealing* algorithm, also called *Metropolis algorithm* [12]. Simulated annealing models the state transition from fluid to crystalline state of metals. From the algorithmic viewpoint, this process is an optimization process with extremely high dimension. To apply simulated annealing to a general optimization problem, one needs to formulate the given problem as a cooling process. For our application, we interpret the configuration of a polygonal surface as the configuration of metal molecules. Our internal energy is represented by a target function, and the random heat movement of molecules is represented by random changes in the configuration. In general, we change a configuration randomly, see Section 4.2., accepting only changes that do not violate the error bound. We compute the new target function, see Section 4.1., and either accept or reject the change, following the rules of simulated annealing. We first applied this approach to the 2D case (simplification of closed polygons) and the good results encouraged us to extend it into 3D.

4.1. The Target Function

The target function describes the quality of an approximation. We experimented with different error norms, but the results were poor. In general, an error norm describes the difference between the original and the approximating surface. However, it is not the goal to minimize this difference but the number of vertices of the approximation. Furthermore, the target function should not only prefer configurations that consist of few vertices, but also configurations that lead to vertex removals. We use the sum of the square roots of the angles between triangle normals as the target function. This function is highly related to the number of vertices. Fewer vertices lead to fewer edges and to fewer angles to be added. This strategy prefers planar surfaces, since a large number of small angles has a higher target function value than a smaller number of large angles. This leads to near-planar platelets of triangles, where we can delete vertices. One could argue that, for planar surfaces, this target function is independent of the vertex number since all angles of neighboring triangles "dihedral angles") are zero. However, in practice a mathematically planar surface cannot be represented by "truly co-planar" triangles due to numerical errors. Furthermore, this target function discourages self-intersections of the surface because self-intersections can only happen in regions of high curvature meaning high angles between triangles. It is also easy to compute and can be recomputed locally after local changes.

4.2. Configuration Changes

To change a configuration, we use the method of Kreylos and Hamann [9], adapted to our problem. In general, we approximate by decimation, like most of the published algorithms [3, 4, 5, 13, 14, 15, 16]. We use three different operations: edge flip, vertex removal, and vertex movement. The edge flip only changes the triangulation of two neighboring triangles. To move a vertex, we randomly choose a new position inside a small sphere around the original one. This enables us, like Hoppe et al. [7], to use more than just original points. However, they do not preserve a global error bound. We then move the vertex to the new position. To check geometric validity of the triangulation, we project all involved triangles onto the plane defined by the vertex normal of the changing vertex. In the case of non-convex platelets, this can lead to degenerate triangles or triangles with wrong orientations. We resolve these conflicts by flipping the appropriate edges. Nevertheless, it is possible that the projection method cannot detect all self-intersections. However, we are not aware of an approach for determining efficiently a plane for projection that is optimal. Our target function punishes self-intersections and therefore our method works well in practice. We store the sequence of the flipped edges in a stack, which allows us to undo the movement with minimal computational effort when the move is ultimately rejected by the simulated annealing scheme. To remove a vertex we collapse the shortest edge emanating from it. Therefore, we do not have to re-triangulate holes. (We have implemented the edge collapse operation by moving one vertex of the edge onto the other, using the standard vertex movement.)

There is a problem with vertex movement: The algorithm requires non-self-intersecting platelets. Yet it is possible to construct platelets where the projection onto the vertex-normal plane does self-intersect. However, these cases are highly unlikely to occur in real data sets and can be neglected for our purposes.

§5. Improvements and Future Research

The main drawback of our algorithm is its lack of computational efficiency. There are two reasons for this: First, tests involving the simplification envelope are expensive. However, considering our goal to satisfy an a priori error bound, this cost cannot be avoided. As mentioned before, even if triangulated representations of the offset surfaces were known, the complexity of the tests would not change significantly. Second, simulated annealing is expensive. On the other hand, a major advantage of simulated annealing is the fact that the random movements provide a mean to use any point inside the envelope as a possible vertex position. However, the large number of necessary movements and the tests involved result in a poor performance. Future work will be done to replace the simulated annealing approach with a more efficient alternative.

The same basic algorithm can also be used with more complicated atomic envelopes. An example is the construction shown in Fig. 2. This construction not only uses the vertex normal, but also the normal of the triangle to create the atomic envelope.

Compared to fundamental prisms, we add three bilinear patches to each atomic envelope, which can all have possible exit points. Furthermore, to test whether an exit point is covered by an atomic envelope is a consequently slower operation. The new atomic envelope embeds the the old one and one or two additional volumes above the top and below the bottom triangles. The additional volumes can also be represented as pentaeder Bézier volumina. However, this new atomic envelope approximates the exact non-linear offset much better, especially in regions of high curvature.

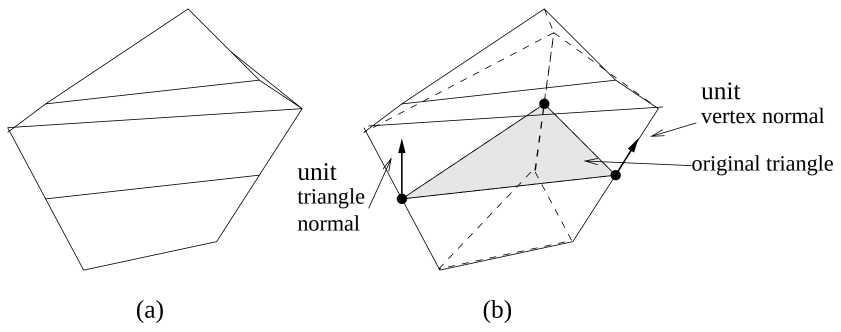

Fig. 2. Different atomic envelopes (a) solid view; (b) transparent view.

	Sphere		Torus		Drill Bit		Cave	
ϵ	Vertices	Time	Vertices	Time	Vertices	Time	Vertices	Time
0	602	0	1000	0	1946	0	1581	0
1/8	540	1	-	-	1589	2	849	1
1/4	358	1	843	1	981	2	611	2
1/2	200	1	493	1	430	2	374	2
1	101	1	231	1	144	3	198	3
3	29	2	72	2	18	9	81	7
5	18	2	39	2	-	-	-	-

Tab. 1. Results.

§6. Results

All simplifications were performed on an SGI Octane with an R10000 processor, running at 250MHz and using 128MBytes of main memory. The two analytical data sets of the sphere and torus were the results of triangulating a parametric representation. The drill bit is available on the web pages of the Department of Computer Science at Stanford University [17]; it is a reconstruction of a laser-range scan. The cave data set was obtained by a range scan using a laser positioned in the center of the cave. For more results and pictures, we refer to [1].

Our absolute error bound, is the given percentage of the average side length of the bounding box of the original model. All times listed in Table 1 are in minutes. In Figures 9 and 10 the true shape is difficult to show, because from all meaningful viewing directions the points scanned from this cave are nearly co-planar.

§7. Conclusions

We have presented an algorithm to simplify closed 2-manifold triangulations. The algorithm constructs a simplified triangulation within an a priori error bound. The concept of atomic envelopes allows us to use any error bound for any surface triangulation. The approach can easily be modified to be applicable to triangulated surfaces with boundaries or to non-manifold surfaces. Our approach can be extended to consider different error bounds in different regions, which would allow adaptive simplifications.

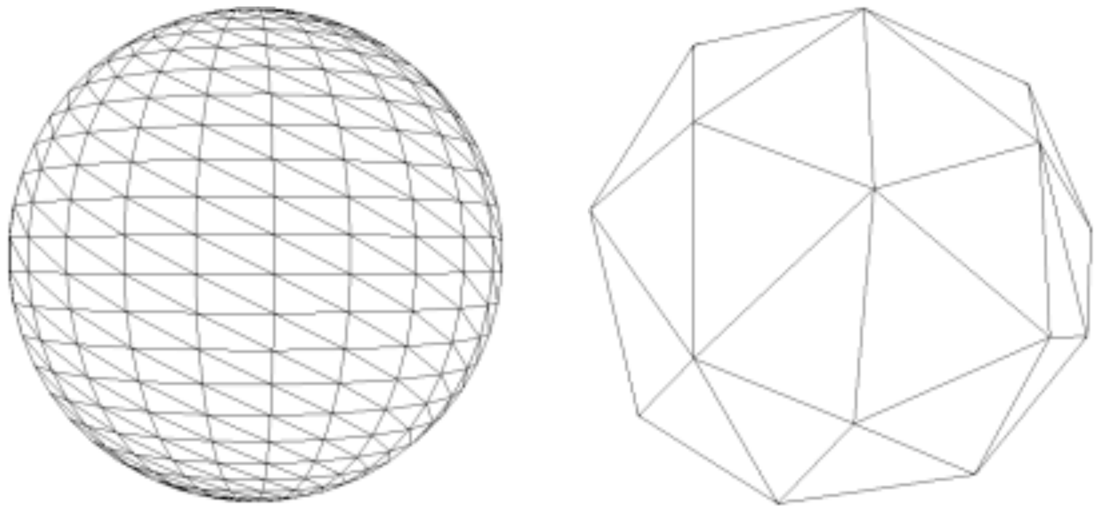

Fig. 3. Sphere data set: (a) original; (b) simplified within 3% error bound.

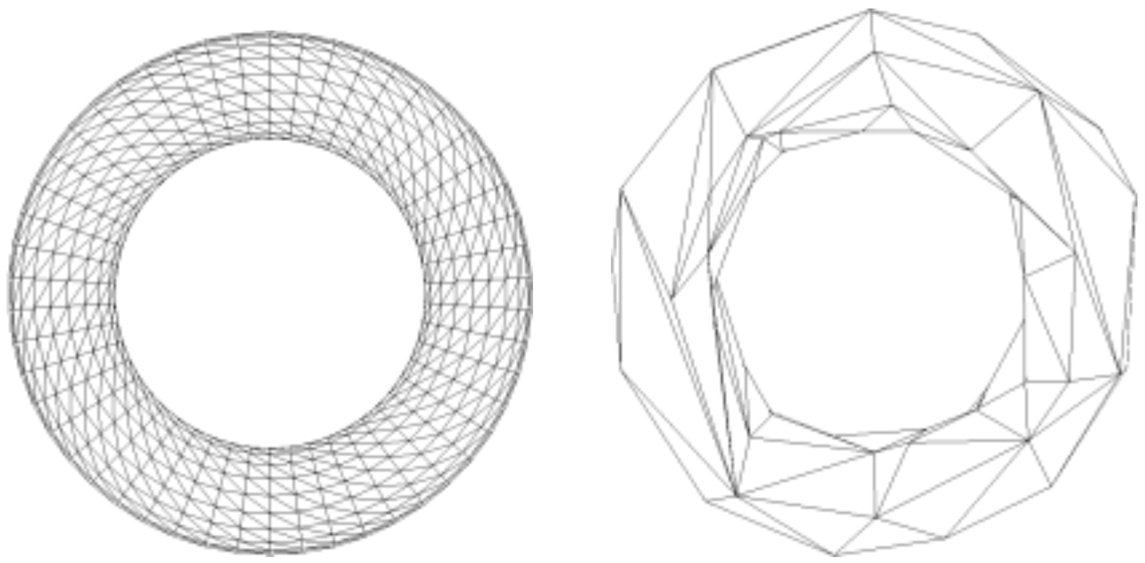

Fig. 4. Torus data set; (a) original; (b) simplified within 3% error bound.

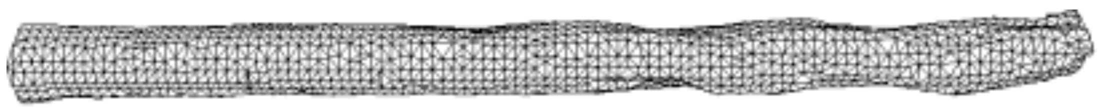

Fig. 5. Original drill bit data set.

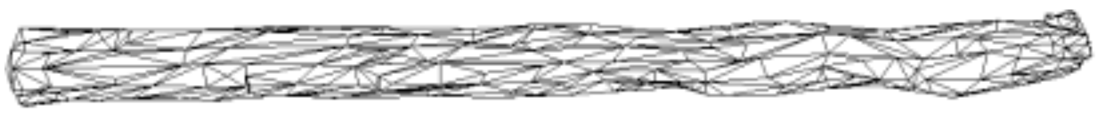

Fig. 6. Drill bit data set simplified within 0.5% error bound.

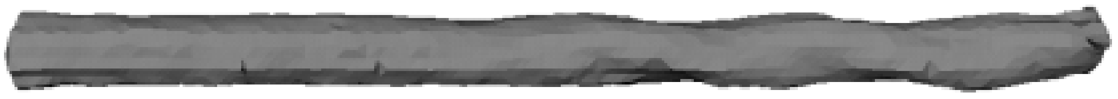

Fig. 7. Original drill bit data set, flatshaded.

Fig. 8. Drill bit data set simplified within 0.5% error bound, flatshaded.

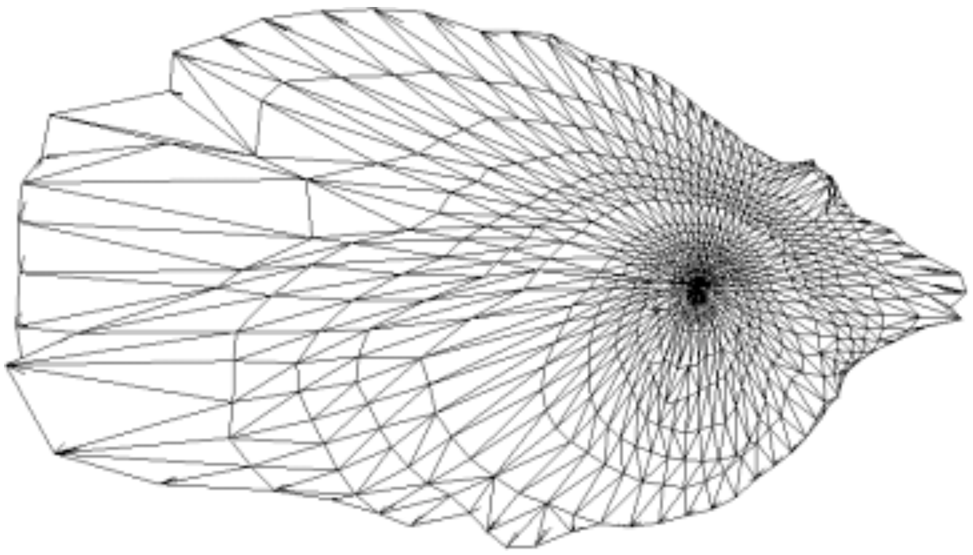

Fig. 9. Original cave data set.

Acknowledgments.

This work was supported by the National Science Foundation under contracts ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the Office of Naval Research under contract N00014-97-1-0222; the Army Research Office under contract ARO 36598-MA-RIP; the NASA Ames Research Center through an NRA award under contract NAG2-1216; the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159; the Lawrence Berkeley National Laboratory; the Los Alamos National Laboratory; and the North Atlantic Treaty Organization (NATO) under contract CRG.971628. We also acknowledge the support of ALSTOM Schilling Robotics and SGI. We thank the members of the Visualization and Graphics Research Group of the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis and the members of the Welfen Laboratory at the University of Hannover, Germany.

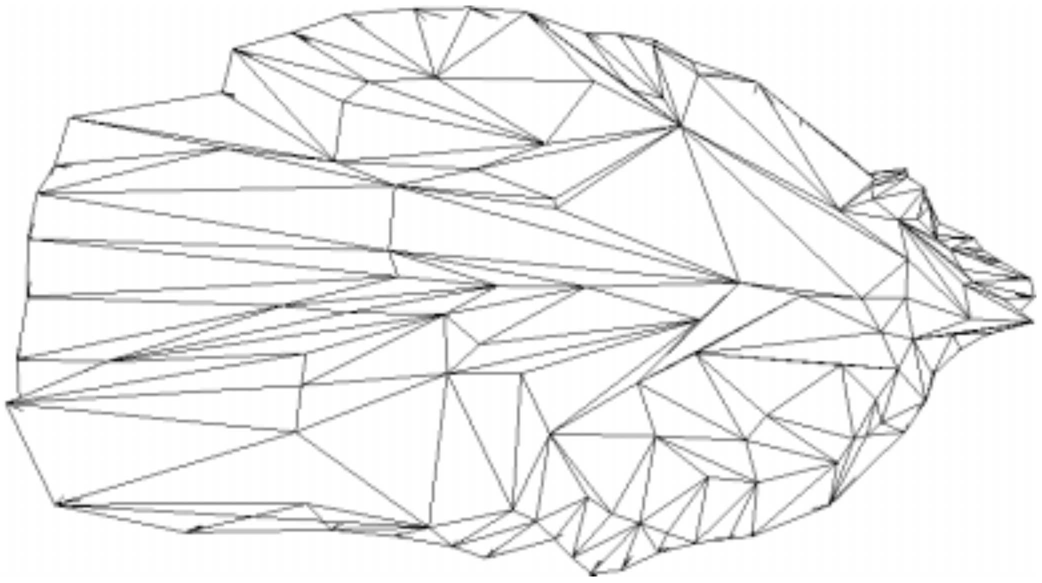

Fig. 10. Cave data set simplified within 1% error bound.

References

1. Bremer, P. T., Boundary simplification of a triangulated body, Diplomarbeit, Fachbereich Mathematik und Informatik, Universität Hannover, 10. Apr. 2000.
2. Cohen, J., A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwal, F. Brooks and W. Wright, Simplification envelopes, Proceedings SIGGRAPH (1996), 119–128.
3. Gourdon, A., Simplification of irregular surface meshes in 3D medical images, Computer Vision, Virtual Reality and Robotics in Medicine, CVRMed 1995, Apr. 1995, 413–419.
4. Guéziec, A., Surface simplification with variable tolerance, Second Annual Intl. Symp. on Medical Robotics and Computer Assisted Surgery, MRCAS 1995, Nov. 1995, 132–139.
5. Hamann, B., A data reduction scheme for triangulated surfaces, Comput. Aided Geom. Design **11**, (1994), 197–214.
6. Heckbert, P. S. and M. Garland, Survey of polygonal surface simplification algorithms, Multiresolution Surface Modeling Course, SIGGRAPH 1997.
7. Hoppe, H., T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle, Mesh optimization, Proceedings SIGGRAPH, Aug. 1993, 19–26.
8. Klopp, S., Implementation von R*-Bäumen als benutzerdefinierte Indexstruktur in Oracle 8i, Studienarbeit, Fachbereich Mathematik und Informatik, Universität Hannover, 1999.
9. Kreylos, O. and B. Hamann, On simulated annealing and the construction of linear spline approximations for scattered data, Proceedings of

the Joint EUROGRAPHICS-IEEE TVCG Symposium on Visualization, Vienna, Austria, May 1999, 189–198.

10. Lasser, D., Bernstein-Bézier-Darstellung trivarianter Splines, Dissertation, Fachbereich Mathematik, Technische Hochschule Darmstadt, Germany, 1987.
11. Lasser, D., Bernstein-Bézier representation of volumes, Comput. Aided Geom. Design **2**, (1985), 145–150.
12. Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, Equations of state calculations by fast computing machines, Journal of Chemical Physics **21** (1953), 1087–1092.
13. Ronfard, J. and J. Rossignac, Full-Range approximation of triangulated polyhedra, Proceedings EUROGRAPIHCS, Computer Graphics Forum **15**, Aug. 1996.
14. Rossignac, J., Interactive exploration of distributed 3D databases over the internet, Proceedings CGI, Hannover, Germany, June 1998, 324–335.
15. Schroeder, W. J., J. A. Zarge and W. E. Lorensen, Decimation of triangle meshes, Proceedings SIGGRAPH, Computer Graphics **26**, (1992), 65–70.
16. Soucy, M. and D. Laurendau, Multiresolution surface modeling based on hierarchical triangulation, Computer Vision and Image Understanding **63** (1996), 1–14.
17. Web pages of the Department of Computer Science at Stanford University, http://www.graph- ics.stanford.edu/data/3Dscanrep/

Peer-Timo Bremer, Bernd Hamann, Oliver Kreylos
Center for Image Processing and Integrated Computing
Department of Computer Science
1 Shields Avenue
University of California
Davis, CA 95616-8562
U.S.A
`tbremer@ucdavis.edu, {hamann,kreylos}@cs.ucdavis.edu`

Franz-Erich Wolter
Lehrstuhl für graphische Datenverarbeitung
Institut für Informatik
Universität Hannover
Welfengarten 1
30167 Hannover
Germany
`few@informatik.uni-hannover.de`