Adaptive Surface Data Compression

Erwin Keeve *, Stefan Schaller **, Sabine Girod ***, Bernd Girod **

Invited paper

* Surgical Planning Laboratory, Dept. of Radiology, Brigham and Women's Hospital, Harvard Medical School
75 Francis Street, Boston, MA 02115, USA, Email: keeve@bwh.harvard.edu

** Telecommunications Institute, University of Erlangen-Nuremberg
Cauerstraße 7, 91058 Erlangen, Germany, Email: {girod, schaller}@nt.e-technik.uni-erlangen.de

*** Department of Oral and Maxillofacial Surgery, University of Erlangen-Nuremberg
Glückstraße 11, 91054 Erlangen, Germany, Email: sabine.girod@mkg.med.uni-erlangen.de

Keywords: medical image compression, surface reconstruction, interactive simulation and visualization

Abstract: Three-dimensional visualization techniques are becoming an important tool for medical applications. Computer generated 3D reconstructions of the human skull are used to build stereolithographic models, which can be used to simulate surgery or to create individual implants. Anatomy-based three-dimensional models are used to simulate the physical behaviour of human organs. These 3D models are usually displayed by a polygonal description of their surface, which requires hundreds of thousands of polygons. For interactive applications this large number of polygons is a major obstacle. We have improved an adaptive compression algorithm that significantly reduces the number of triangles required to model complex objects without losing visible detail and have implemented it in our surgery simulation system. We present this algorithm using human skull and skin data and describe the efficiency of this new approach.

Zusammenfassung: Computerbasierte dreidimensionale Visualisierungstechniken haben im letzten Jahrzehnt Einzug in die Medizin gehalten. Aus den computergenerierten dreidimensionalen Rekonstruktionen des Gesichtsschädels werden unter anderem mittels Stereolithographie reale Modelle erstellt, an denen geplante chirurgische Eingriffe simuliert werden können, oder aber die 3D-Rekonstruktionen dienen dazu, patientenangepaßte Implantate herzustellen. Die Geometrie solch komplexer 3D Modelle wird im allgemeinen mit Hilfe hunderttausender einzelner, planarer Polygone beschrieben. Eine interaktive Darstellung dieser Modelle ist oftmals nicht mehr möglich. In dieser Arbeit beschreiben wir ein erweitertes adaptives Verfahren zur signifikanten Reduzierung von Polygonoberflächen, ohne daß damit ein Detailverlust in der Darstellung verbunden ist. Dieses Reduzierungsverfahren wurde in ein Operationsplanungssystem integriert und umfassend verifiziert. An zwei medizinischen Datensätzen, der 3D Rekonstruktion der Hautoberfläche und des Gesichtsschädels, wird die Leistungsfähigkeit dieses neuen Verfahrens aufgezeigt.

1. Introduction

Computer-based three-dimensional visualization techniques have made a great impact on the field of medicine in the last decade *[Zon94]*. Physical models of the skull can be created through computer-generated reconstruction *[Bil95]*, individual implants can be produced *[Weh95]*, or surgical operations can be simulated using state-of-the-art graphics workstations *[Kee96a] [Kee96b] [Kee96c] [Kee96d]*. Such modern visualization methods also allow the use of surgery robots that help with the exact positioning of surgical instruments, especially in neurosurgery *[Haß95]*.

The visualization of such three-dimensional objects takes place by using volume or surface rendering techniques *[Fol92]*. Volume rendering - the direct rendering of data represented as 3D scalar fields - is becoming an important branch of computer graphics. It allows the visualization of three-dimensional transparent datasets without obscuring the interior of objects. Although there exists several compression algorithms for volume visualization *[Nin92] [Nin93]* there is still an even greater need for hardware speed than with surface rendering, since volume datasets are generally much larger.

In most cases medical objects are visualized using surface rendering techniques. Using polygonal primitives, the object surface is described and visualized through common computer graphics hardware. Since a single polygon is planar, a large number of primitives is required to capture the detail of complex, curved objects such as anatomical structures. One algorithm that extracts isodensity surfaces from volume data is *"Marching Cubes" [Lor87]*. Developed for medical applications, *"Marching Cubes"* generates typically 500,000 to 1,000,000 triangular primitives from a 512 by 512 by 100 CT-dataset of an anatomical object, such as a skull. Sampling devices such as a Cyberware 3D-laser scanner obtain models of the human head using 500,000 triangles. This large number of polygons slows down the rendering and inhibits real-time interaction. Even today's graphics workstation have trouble storing and rendering models of this size. Therefore a surface compression technique is required which speeds up the rendering while preserving the visible detail of such anatomy-based models.

In this paper we present an adaptive surface data compression algorithm which classifies the surface primitives by their topology, geometry, neighbouring normals and user-specified criteria. After outlining other work related to surface data compression, we describe our approach and its implementation in detail. Results show how this technique reduces human skin and skull data without sacrificing the visible detail of the anatomical models.

2. Related Work

There are two main categories in surface data compression; filter-based and adaptive approaches.

Filter-based techniques are well-known and proceed on a large number of samples to remove or replace them. Examples of this technique are sub-sampling and averaging. Sub-sampling uses every n'th sample, while averaging combines neighbouring primitives to reduce the size of the dataset.

Adaptive techniques reduce primitives only if specified criteria are satisfied. For example, Yoshida et al. decimate a triangle mesh in smooth areas by degenerating one edge of a triangle toward a point *[Yos93]*. Schmitt starts with rough bi-cubic patch approximations to sample data, then subdivides those patches that are not sufficiently close to the underlying samples *[Sch86]*. De Hämer extends this technique to reduce polygonal meshes *[DeH92]*. Turk randomly places a given number of vertices on a polygonal mesh and then specifies a new mesh by retriangulation *[Tur92]*. To yield the topological structure of the original mesh, all new vertices are first introduced to the existing mesh and then the original samples are discarded. Deformable models are another adaptive technique to resample a given dataset *[Mil91]*. An initial surface model is deformed until it fits the implicit surface that exists within a sampled volume. Also adaptive meshing techniques *[Ter91]* that are employed in *[Wat95]* are used to reduce the large data acquired by a 3D-laser scanner into a parsimonious geometric model of the face that can be animated efficiently *[Lee95]*. To create new views of 3D models from arbitrary camera positions Levoy and Hanrahan developed a method which simple combines and resamples the available images *[Lev96]*.

Another adaptive compression technique, which considers the local topological and geometrical structure of a polygonal mesh, was developed by Schroeder, Zarge and Lorensen, in 1992 *[Sch92]*. The presented surface data compression approach is based on the work by Schroeder et al. and improves this technique for medical applications *[Sch95]*.

3. Compression Algorithm

The goal of the presented adaptive compression algorithm is to reduce the amount of triangles used to represent complex anatomical structures without sacrificing the visible detail of the models. This is done by using a subset of the original vertices. There is no creation of new vertices; instead vertices which meet user-specified criteria are removed from the dataset.

Starting by classifying the surface primitives by their topology, geometry and their normals, an indicator is set for each primitive vertex whether it can be considered for removal or not. If a marked vertex meets the user-specified compression criterion, the vertex and all primitives that use it, are deleted. A new local triangulation with fewer primitives is formed to patch the resulting "hole". This process is repeated until a termination criterion is satisfied.

3.1 Data Aquisition

The compression algorithm is used in a surgery planning system which simulates craniofacial operations *[Kee96a] [Kee96d]*. This system obtains data from a Cyberware scanner, which measures exactly the geometry of the skin surface *[Cyb93]*. By providing colour information as well, this 3D laser scanner makes an additional contribution to the photorealistic appearance of the patient's face. The system also requires skull data, which are generated through computer tomography. The *"Marching-Cubes"* method is used to build a 3D model from this CT dataset *[Lor87]*.

3.2 Data Structure

There are many approaches to represent polygonal data structures *[Wei85]*. Because we have a large amount of data, the chosen data structure must be efficient, while still providing the functionality that we need. It must allow the efficient retrieval of all vertices of a given primitive, the finding of all primitives of a given vertex and the finding of all neighbouring vertices of a given vertex. The only primitives being used in our approach are triangles, since they capture the detail of complex, curved objects well and can be rendered efficiently using common graphics hardware. The brute force approach would be to save a list of all triangles and their vertex coordinates, but this is inefficient, because the coordinates of shared vertices are multiply stored. Another approach lessens storage space by only storing the vertex coordinates once and maintaining a primitive list with references to these coordinates. This index list is efficient but does not easily provide the above functionality. A more useful data structure is the *Winged-Edge* representation by Baumgart *[Bau75]*, which keeps a list of all primitive edges and therefore is not as efficient.

Our implementation uses a space-efficient vertex-triangle hierarchical ring structure *[Sch92]*. This consists of a list of all triangles whose indices refer to a vertex coordinate array. Connections are built from the coordinate array back to the triangle list by lists of triangles using each vertex. Additionally another list is maintained that contains an index of all neighbouring vertices for each vertex. Edge definitions are not explicitly defined, instead they are implicitly given by the order of the vertex indices in the triangle list. Figure 1 shows this data structure.

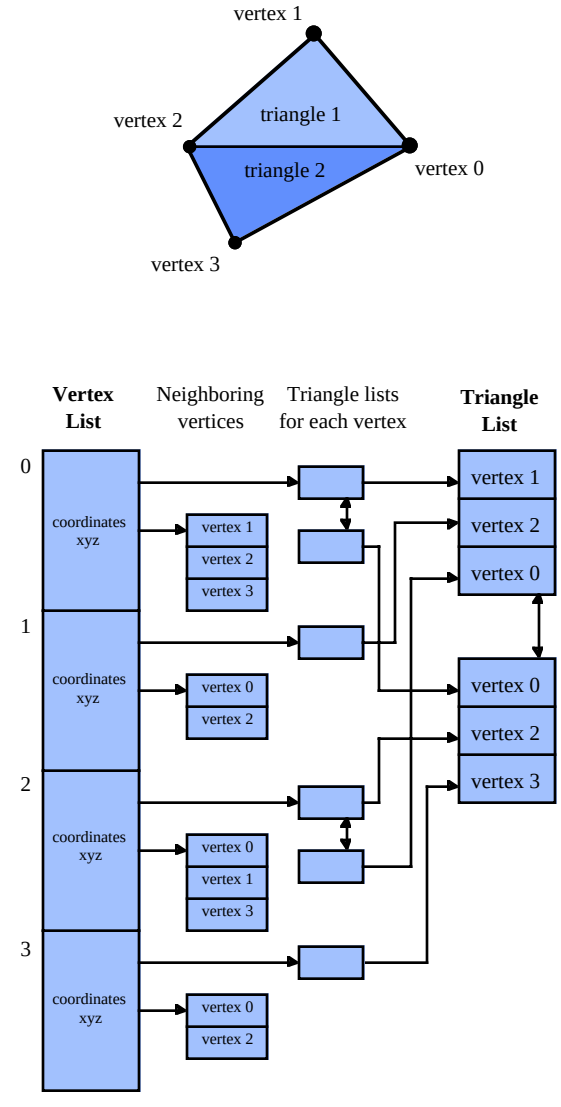

Figure 1: Implemented data structure

3.3 Classification

Each vertex is classified by three criteria; by its topology, by its geometry and by its neighbouring normals. We only continue to consider vertices that successfully pass each successive criterion. After the topological and geometrical classification steps there are five possible cases; two of which are never allowed to be decimated, and three of which proceed to further normal testing. If the normal criterion is also successful, then user-specified criteria are applied. Figure 2 shows an overview of all classification steps.

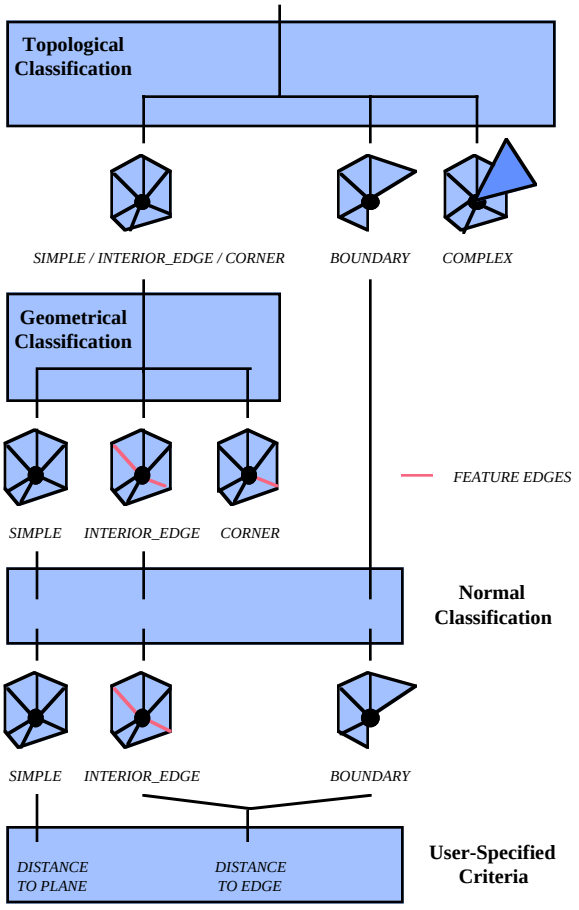

Figure 2: Classification

Topological Classification: The first step is to classify each vertex according to the topology of its neighbouring triangles. There are three possible topological classifications; *COMPLEX*, *BOUNDARY* and *SIMPLE / INTERIOR_EDGE / CORNER*. A triangle is called two-manifold if its edges belong to at most two primitives *[Fol92]*. Under the topological classification, a vertex whose triangle is not two-manifold is labeled as *COMPLEX* and is not further considered for reduction. Figure 3 shows two *COMPLEX* vertices. *BOUNDARY* vertices, which are those that are not completely surrounded by triangles, do not pass through the geometry classification, but proceed to further normal and user-specified testing. All vertices which are completely surrounded by triangles are grouped together as *SIMPLE / INTERIOR_EDGE / CORNER* and will be further classified according to geometry.

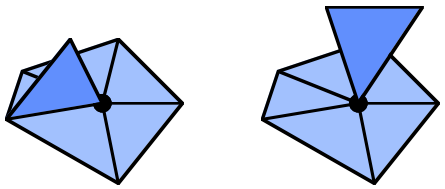

Figure 3: *COMPLEX* vertices

Geometrical Classification: The *SIMPLE / INTERIOR_EDGE / CORNER* vertices are further separated into three categories; *SIMPLE, INTERIOR_EDGE* and *CORNER*. Each triangle edge is examined with respect to the angle between the normals of its adjacent triangles. This edge is labeled as a *FEATURE EDGE* if the angle exceeds a given user-specified *FEATURE EDGE ANGLE*, as shown in Figure 4. Each vertex is classified by the number of *FEATURE EDGES* which are connected to it. If a vertex has no connecting *FEATURE EDGES* then it is labeled as *SIMPLE*. If it has two connecting *FEATURE EDGES* it is labeled as *INTERIOR_EDGE*. Otherwise, it is labeled *CORNER* and is not considered for reduction. *SIMPLE* and *INTERIOR_EDGE* vertices will further proceed to normal and user-specified testing.

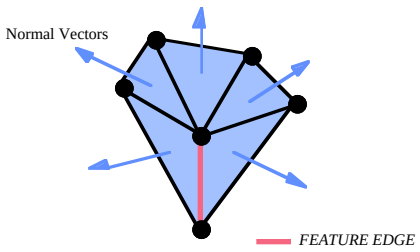

Figure 4: *FEATURE EDGE*

Normal Classification: Because normals have a great impact on the appearance of rendered anatomical structures, an additional normal criterion is defined. All *SIMPLE*, *INTERIOR_EDGE* and *BOUNDARY* vertices are analysed with respect to the normals of their neighbouring vertices; if the angle between the normal of each vertex and its adjacent vertices is larger than a user-specified *NORMAL VARIATION ANGLE*, the vertex is not considered for removal. Otherwise it is passed through to the final user-specified testing.

3.4 User-Specified Criteria

After assigning each vertex to one of *SIMPLE*, *INTERIOR_EDGE*, *CORNER*, *BOUNDARY* or *COMPLEX* categories, user-specified decimation criteria can be defined for each class. To keep the topological structure of the mesh, *COMPLEX* vertices are never considered for removal. Also *CORNER* vertices are not reduced, in order to preserve sharp edges.

SIMPLE vertices are tested using the user-specified *DISTANCE TO PLANE* criterion, as shown in Figure 5. An *AVERAGE PLANE* is constructed from the neighbouring vertices. If the distance from the vertex to the *AVERAGE PLANE* is greater than the user-specified value, the vertex is retained. If the distance is less, then the vertex can be considered for removal.

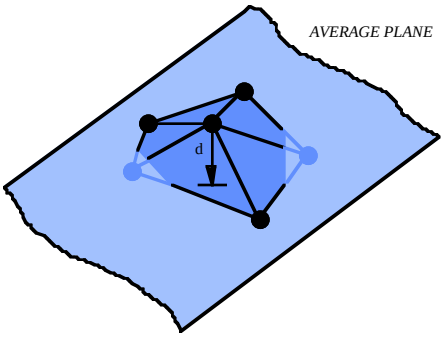

Figure 5: *DISTANCE TO PLANE* criterion

INTERIOR_EDGE and *BOUNDARY* vertices are tested by the user-specified *DISTANCE TO EDGE* criterion, as shown in Figure 6. By definition, these vertices are connected to two *FEATURE EDGES* or they build part of the boundary. The distance from the vertex to the line defined by the pair of vertices forming the *FEATURE EDGES* or to the line defined by the two other vertices forming the boundary segment, is determined. If this distance is less than the user-specified value, the vertex can be considered for removal and otherwise it is retained.

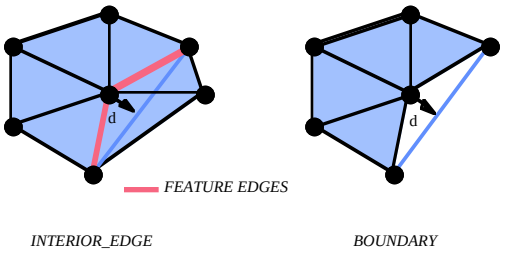

Figure 6: *DISTANCE TO EDGE* criterion

3.5 Retriangulation

After these classification steps, each vertex is now marked with an indicator whether it could be removed or not. In a further step we consider the submesh around each positively marked vertex and attempt to retriangulate the remaining polygons without it. If the retriangulation is not successful, we must retain the vertex and its triangles.

If we remove a *SIMPLE* or *BOUNDARY* marked vertex from a submesh, we are left with one remaining polygon, which must be retriangulated. However, if we remove an *INTERIOR_EDGE* marked vertex we are left with two remaining polygons, which both must be retriangulated. Figure 7 shows this.

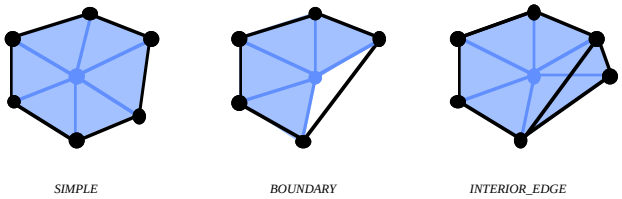

Figure 7: Remaining polygons

When we retriangulate the remaining polygons we have two goals to take into consideration; the new triangulation must approximate the original shape well and each triangle should be within a user-specified *TRIANGLE ASPECT RATIO*. To do this, we split each remaining polygon into two halves. This splitting procedure is recursively repeated until each half has only three vertices forming a new triangle. If we could not split the remaining polygon into new triangles within an user-specified aspect ratio, the vertex under consideration and all of its triangles must be retained.

The remaining polygon can be split along a few possible lines. We enumerate all of the possible *SPLIT LINES* between pairs of non-neighbouring vertices and build *SPLIT PLANES* along these lines orthogonal to the *AVERAGE PLANE*, as shown in Figure 8. For each possible *SPLIT PLANE*, we calculate the distance from each vertex to it and take the ratio of the smallest distance to the length of the *SPLIT LINE*. The *SPLIT LINE* with the greatest ratio will deliver new triangles with the best aspect ratio.

As it follows from the Euler relation the removal of a *SIMPLE* or a *INTERIOR_EDGE* vertex reduces the submesh by two triangles *[Pre88]*. Removal of a *BOUNDARY* vertex reduces the submesh by one triangle.

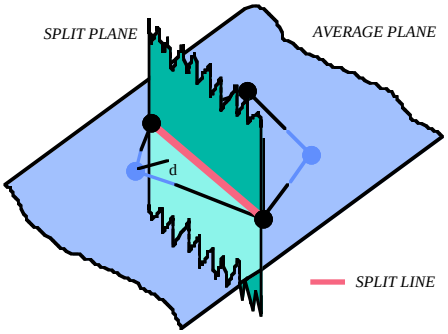

Figure 8: *SPLIT PLANE*

After all vertices have passed through these different decimation steps, the loop is started again until a user-specified *NUMBER OF ITERATIONS* is reached.

3.6 Special Cases

There exists several special cases in general, which must be taken into account. However, because we are using only triangle meshes generated by the *"Marching-Cubes"* algorithm or directly reconstructed from Cyberware scanner data, attention must be given to only two special cases during retriangulation.

- If there is a topological hole in the triangle mesh it might be possible to remove a vertex from this hole's boundary. This could cause the hole to collapse and would change the topology.
- Also for a simple closed surface such as a tetrahedon removing of one vertex changes the topology of the mesh.

To prevent this, checks are made during retriangulation to insure that dublicate triangles or triangle edges are not generated.

3.7 Software Implementation

The algorithm is part of a surgical simulation system which is implemented in C++ on Silicon Graphics workstations, using the object-oriented 3D graphics library Open Inventor *[Wer94]*. The user-interface was created with the Motif library and is shown in Figure 9. It provides flexibility by allowing the user to set the following parameters

- *DISTANCE TO PLANE*
- *DISTANCE TO EDGE*
- *FEATURE EDGE ANGLE*
- *TRIANGLE ASPECT RATIO*
- *NORMAL VARIATION ANGLE*
- *NUMBER OF ITERATIONS*

Additionally, to simplify the user interaction, an *OVERALL QUALITY LEVEL* is provided which takes reasonable default parameter values.

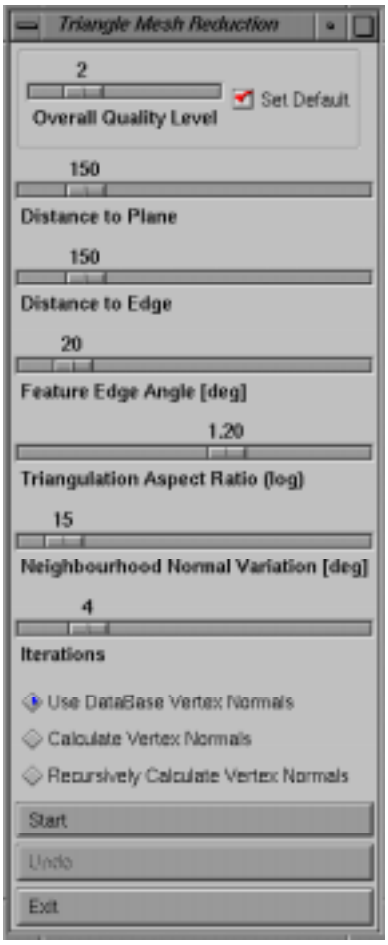

Figure 9: User-interface

4. Results

As described above, in our surgical planning system we use two different types of data - Cyberware scanner skin data and *"Marching-Cubes"* reconstructions of the skull from computer tomography. Although the initial meshes are generated by different methods, all results are produced by the same compression algorithm.

As shown in Figure 10 on a extract of human skin data, our algorithm reduces the original dataset from 46,000 to 5,390 triangles (compression ratio 8.5:1) without sacrificing the visible detail of the model. Texture mapping further improves the appearance. The differences between the reduced and original dataset are visualized in the lower row. The maximal error for the

compression rate 2:1 is 0.25 mm and for the compression rate 8.5:1 is 0.4 mm. The compression of the skin data takes less than one minute on a common graphics workstation, such as a SGI Indigo High Impact. Afterwards, the reduced dataset can be used interactively in our surgery simulation system running on the same platform. This would not be possible without compression.

Figure 11 shows the compression of skull data. The algorithm reduces the original dataset from 343,696 to 95,808 triangles (compression ratio 3.5:1). In this application the compression ratio is less than in the above example, because the surface structure in the area of the teeth and vertebrae is more complex and highly curved.

We compared our improved method (with normal variation) with the original approach developed by Schroeder et al. (without normal variation). As shown in Figure 12 on a extract of human skin data the surface looks more smooth with the improved method.

5. Conclusions

We have improved an adaptive compression algorithm developed by Schroeder et al. by adding an additional normal classification step to it. The implemented algorithm can reduce the number of triangles required to model complex objects by approximately 8:1 without losing visible detail. We have implemented this algorithm and it enables interactive manipulation of complex anatomical objects, which is a fundamental prerequisite of a surgical simulation system. The interface gives users the flexibility to easily tailor parameters to suit the needs of their application. The presented results demonstrate the efficiency and strengths of this approach.

Acknowledgement: This work is supported by Deutsche Forschungsgemeinschaft (DFG) research grant Gi 198/2-4 and by German Academic Exchange Service (DAAD) research grant D/96/17570.

This software is available via internet under http://splweb.bwh.harvard.edu:8000/pages/ppl/keeve

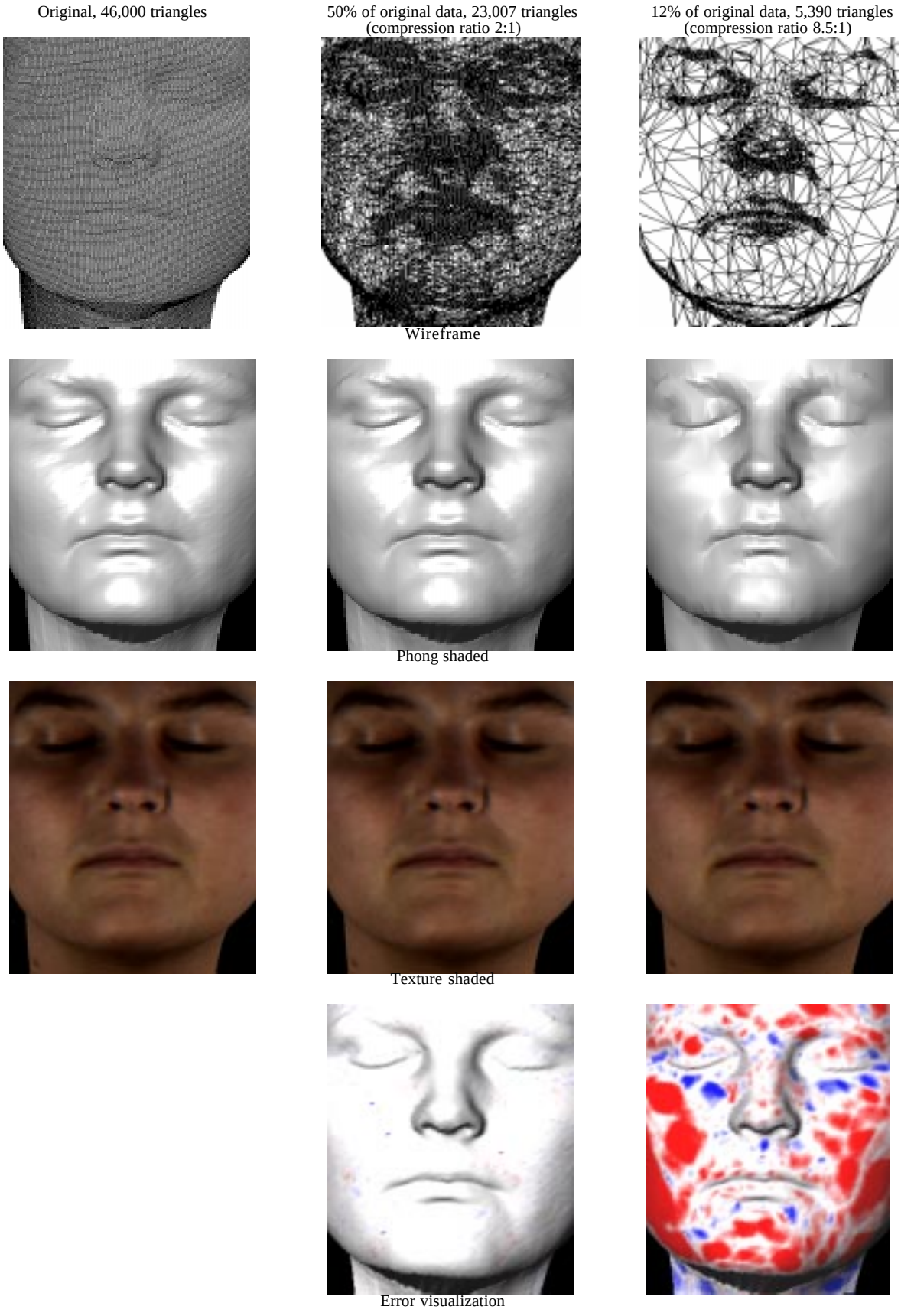

Figure 10: Skin data

Figure 11: Skull data

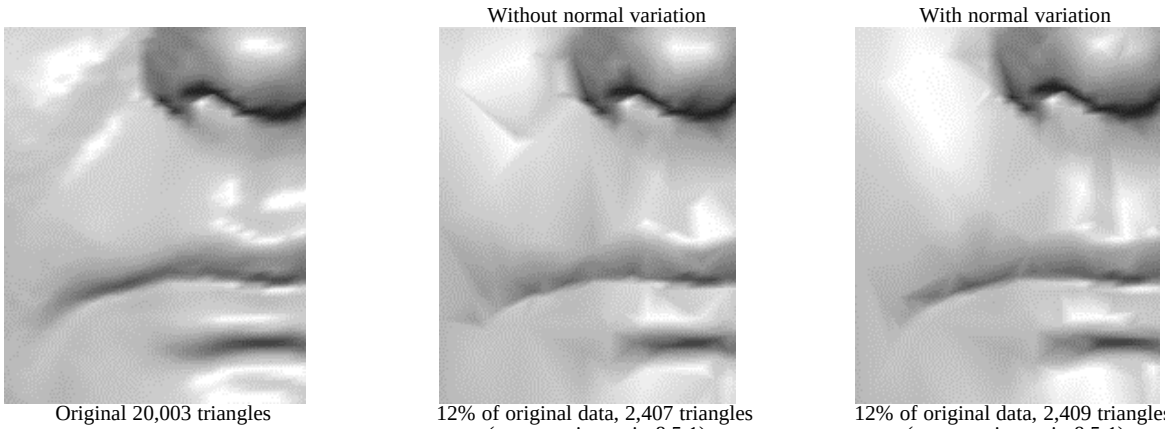

Figure 12: Comparison

References:

[Bau75] BAUMGART B. G., *"Winged Edge Polyhedron Representation,"* Technical Report STAN-CS-320 Computer Science Department, Stanford University Palo Alto, CA, 1972.

[Bil95] BILL J. S., REUTHER J. F., DITTMANN W., KÜBLER N. MEIER J. L., PISTNER H., WITTENBERG G. *"Stereolithography in Oral and Maxillofacia Operation Planning,"* Int. J. Oral Maxillofac. Surgery Volume 24, No. 1, Part II, pp. 98-101, 1995.

[Cyb93] CYBERWARE LABORATORY INCORPORATED (ed.) *"Cyberware Model 3030 Digitizer Manual,"* Monterey CA, 1993.

[DeH92] DE HÄMER M. J., ZYDER M. J., *"Simplification o Objects Rendered by Polygonal Approximations,"* Computers & Graphics, Vol. 15, No. 2, pp. 175-184 1992.

[Fol92] FOLEY J. D., VAN DAM A., FEINER S., HUGHES J. F. *"Computer Graphics - Principles and Practice,"* Addison-Wesley, Readings, MA, 1992.

[Haß95] HAßFELD S., MÜHLING J., ZÖLLER J., *"Intraoperative Navigation in Oral and Maxillofacial Surgery,"* Int. J Oral Maxillofac. Surgery, Volume 24, No. 1, Part II pp. 111-116, 1995.

[Kee96a] KEEVE E., GIROD S., GIROD B., *"Computer-Aided Craniofacial Surgery,"* Proc. Computer Assisted Radiology CAR'96, Paris, France, June 26-29, pp. 757-762, 1996.

[Kee96b] KEEVE E., GIROD S., GIROD B., *"Craniofacial Surgery Simulation,"* Proc. 4th International Conference on Visualization in Biomedical Computing VBC'96 Hamburg, Germany, Sept. 22-25, pp. 541-546, 1996.

[Kee96c] KEEVE E., GIROD S., PFEIFLE P., GIROD B., *"Anatomy-Based Facial Tissue Modeling Using the Finite Element Method,"* Proc. IEEE Visualization'96, San Francisco, CA, Oct. 27 - Nov. 1, 1996.

[Kee96d] KEEVE E., *"Visualisierungs- und Simulationsverfahren zur interaktiven Planung kraniofazialer Korrekturoperationen,"* Ph.D. thesis, University of Erlangen, 1996.

[Lee95] LEE Y., TERZOPOULOS D., WATERS K., *"Realistic Modeling for Facial Animation"*, Proceedings ACM SIGGRAPH'95, 1995.

[Lev96] LEVOY M., HANRAHAN P., *"Light Field Rendering"* Proceedings ACM SIGGRAPH'96, pp. 31-42, Aug 1996.

[Lor87] LORENSEN W. E., CLINE H. E., *"Marching Cubes: A High Resolution 3D Surface Construction Algorithm,"* ACM Computer Graphics, Vol 21, No. 4, pp. 38-44 July 1987.

[Mil91] MILLER J. V., BREEN D. E., LORENSEN W. E., O'BARA R. M., *"Geometrically Deformable Models: A Method for Extracting Closed Geometrical Models from Volume Data,"* ACM Computer Graphics, Vol. 25, No 3, July 1991.

[Nin92] NING P., HESSELINK L., *"Fast Volume Rendering of Compressed Data,"* Proc. ACM Volume Visualization Workshop, Boston, MA, Oct. 19-20, pp. 69-74, 1992.

[Nin93] NING P., HESSELINK L., *"Vector Quantization for volume rendering,"* Proc. IEEE Visualization '93, San Jose, CA, pp. 11-18, Oct. 1993.

[Pre88] PREPARATA F. P., SHAMOS M.I., *"Computational Geometry - An Introduction,"* Springer Verlag, New York, 1988.

[Sch95] SCHALLER S., *"Ein Beitrag zur Implementierung eines interaktiven Operationsplanungssystems,"* Diplomarbeit, Universität Erlangen-Nürnberg, Januar 1995.

[Sch86] SCHMITT F. J., BARSKY B. A., DU W., *"An Adaptive Subdivision Method for Surface-Fitting from Sampled Data,"* Computer Graphics, Vol. 20, No. 4, pp. 179-188, August 1986.

[Sch92] SCHROEDER W. J., ZARGE J. A., LORENSEN W. E. *"Decimation of Triangle Meshes,"* Proceedings ACM Computer Graphics, Vol. 26, No. 2, pp. 65-70, July 1992.

[Ter91] TERZOPOULOS D., WATERS K., *"Techniques for Realistic Facial Modeling and Animating,"* Computer Animation, Springer, pp. 59-73, 1991.

[Tur92] TURK G., *"Re-Tiling Polygonal Surfaces,"* ACM Computer Graphics, Vol. 26, No. 2, pp. 55-64, July 1992.

[Wat95] WATERS K., FRISBIE J., *"A Coordinated Muscle Model for Speech Animation,"* Proceedings of Graphics Interface '95, pp. 163-170, Mai 1995.

[Weh95] WEHMÖLLER M., EUFINGER H., KRUSE D., MAßBERG W. *"CAD by Processing of Computer Tomography Data and CAM of Individually Designed Protheses,"* Int. J Oral Maxillofac. Surgery, Volume 24, No. 1, Part II pp. 90-95, 1995.

[Wei85] WEILER K., *"Edge-Based Data Structures for Solid Modeling in Curved-Surface Environments,"* IEEE Computer Graphics and Applications, pp. 21-40 January 1985.

[Wer94] WERNECKE J., *"The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor,"* Addison-Wesley, Reading, MA, 1994.

[Yos93] YOSHIDA R., MIYAZAWA T., DOI A., OTSUKI T:, *"Clinical Planning Support System - CLIPSS,"* IEEE Computer Graphics and Applications, November 1993.

[Zon94] ZONNEVELD W., *"A Decade of Clinical Three-Dimensional Imaging: A Review, Part III: Image Analysis and Interaction, Display Options, and Physical Models,"* Investigative Radiology, Volume 29, No. 7, pp. 716-725, 1994.