INTERACTIVE VISUALIZATION OF MULTIDIMENSIONAL DATA

CHANDRAJIT L. BAJAJ

Department of Computer Science, Purdue University,

West Lafayette, IN 47907, USA

bajaj@cs.purdue.edu

Abstract

Interactive data visualization concerns the real time manipulation of sampled and computed data for comprehensive display. The goal of the visualization is to bring to the user a deeper understanding of the data as well as the underlying physical laws and properties. In this survey paper I focus on two problem areas relating to interactive data visualization. The first is in data reduction and dealing with the computation of error-bounded reduced detail meshes with topological correctness. The second is in data reconstruction from scattered point scans and dealing with accurate spline models which support interactive querying and manipulation. For each of these problem areas I survey some of the related research and simultaneously present some of our own recent results.

1 Introduction

Interactive data visualization concerns the real time manipulation of sampled and computed data for comprehensive display. The goal of the visualization is to bring to the user a deeper understanding of the data as well as the underlying physical laws and properties. Such visualization may be used to enlighten a physicist on the complex interaction between electrons, to guide the medical practitioner in surgery, or simply to fly over the surface of a planet which has never been seen by human eyes.

Through the presentation of massive amounts of data as images, we allow the visualization user to rapidly prune useless information, focus on necessary information, and comprehend the science behind the data. Interaction with data brings another level of understanding. Static images can be misleading and mask important features of the data. Motion in visualization brings out hidden features which are inherently dynamic. Interactive manipulation and control of visualization is an important tool which allows scientists to more quickly focus on the region of interest. In environments which are immersive, the motion is critical, to the point that delays or inconsistencies can make the viewer ill. In

this case there is a desire to bound response time using time-critical techniques. Here we consider user interaction on the workstation, although the speed-up techniques can prove useful in other real-time applications as well.

The important aspects of interactive data visualization can be broken down into three categories:

Computation - the ability to speedily compute a visualization. This may include computing a spline approximation to an isosurface of a scalar function, or the computation of a particle trace through a time-dependent vector field, or any action which requires extracting an abstract object or representation from the data being examined.

Display - the ability to quickly display the computed visualization. Display encompasses both computed visualizations as listed above, as well as projection display methods such as volume visualization and ray tracing.

Querying - the ability to interactively probe a displayed visualization for both quantitative and metric information with the purpose of further understanding on a fine scale what is being displayed on a large scale.

Previous research for interactive visualization has focused on each of the three basic needs: computation, display, and querying of scientific data. Algorithmic techniques developed can be categorized as *Reducing Detail*, *Reducing Computation* and *Spatial Data Structures*. In this survey paper I focus on two algorithmic areas relating to interactive data visualization. The first is in data reduction and dealing with the computation of error-bounded reduced detail meshes with topological correctness. The second is in data reconstruction from scattered point scans yielding accurate spline data structures which support interactive querying and manipulation. For each of these areas I survey some of the related research and simultaneously present some of our own recent results.

2 Data Reduction

Interactive visualization is often impeded by the sheer size of the data being visualized. One approach for reducing this stress is through reduction in the number of primitives used to represent the data with a specified level of fidelity. Reduction of detail can refer to elimination of geometric detail, data detail, or both. Reduction of geometric detail is commonly used in real-time environments and simulations. Interactive frame rates are critical in such applications as flight simulators, Virtual Reality walk-throughs, etc. The goal of geometry reduction is to produce a reduced detail representation for a given object, based on certain criteria which, hopefully, provide a measure of the error between the two representations.

Algorithms for reducing the complexity of models can be placed into three categories:

Vertex insertion/deletion - algorithms which iteratively delete vertices from a dense representation, or add vertices to a sparse representation, until a satisfactory result is achieved.

Optimization - algorithms which iteratively converge to a desirable result through use of energy minimization.

Multi-resolution analysis - algorithms which use a functional decomposition of data into nested levels of resolution in order to isolate the important features in the data.

2.1 *Vertex insertion/deletion*

Vertex insertion and deletion forms the basis for many mesh reduction algorithms. In the case of arbitrary surface meshes (geometric mesh reduction), vertex deletion is the most natural approach. For functional surfaces, both insertion and deletion techniques have been considered.

Schroeder, et al [38], compute reduced representations for dense triangular surface meshes such as those computed by Marching Cubes[33] or similar isosurfacing algorithms. Vertices in the dense mesh are examined and classified based on geometric features in the triangulation surrounding the vertex. If error criteria are satisfied, the vertex is deleted and the resulting hole is retriangulated. There is no error propagation, and therefore no guarantee on the amount of accumulated error in the final representation.

Hinker, et al [31] perform "geometric optimization" on triangular surface meshes be grouping faces into contiguous sets which are nearly co-planar, deleting points which are interior or along nearly linear edges, and retriangulating the resulting hole. They note that vertex deletion methods such as this have the advantage that auxiliary vertex information (normals, texture coordinates) can be propagated to the resulting reduced mesh, however their method fails to consider such auxilary information in the vertex removal criteria, opting to consider only the geometric aspect of the mesh reduction.

Much of the work for functional surfaces has been related to mesh reduction for dense geographical data. Lee [4] contains an overview and comparison of point insertion and point deletion methods for functional surfaces. In this context, the function is that of height, and the goal is to accurately model a terrain using the fewest number of elements possible. Geographical Information Systems (GIS) researchers have adopted the Triangulated Irregular Network (TIN) as a representation for simplified meshes [29]. Exploiting the nature of the data defined over a planar mesh, exact errors may be computed at the original mesh points as a measure of error.

2.2 *Optimization*

Optimization methods define measures of energies for point sets or triangulations based on an original mesh, and attempt to minimize these energies in forming a simplified mesh.

In Turk [40], reduced polygonal surfaces are computed at a desired resolution of vertices. Contrast this the point insertion and deletion methods which are driven by error computations rather than desired resolution. Given the desired number of vertices, point repulsion on the polygonal surface spreads the points out. A mutual tessellation of the original triangulation and the introduced points followed by deletion of the original vertices guarantees that the topology of the polygonal surface is maintained. Point repulsion is adjusted based on estimated curvature of the surface, providing an adaptive triangulation which maintains geometric features.

Hoppe, et al. [32] perform time-intensive mesh optimization based on the definition of an energy function which balances the need for accurate geometry with the desire for compactness in representation. The level of mesh reduction is controlled by a parameter in the energy function which penalizes meshes with large numbers of vertices, as well as a spring constant which helps guide the energy minimization to a desirable result.

2.3 *Multiresolution analysis*

Multiresolution analysis is a structured mathematical decomposition of functions into multiple levels of representation. Through the use of wavelet transforms [35, 28], a hierarchical representation of functions can be obtained by repeatedly breaking the function into a coarser representation in addition to a set of perturbation coefficients which allow the full recovery of the original representation from the coarse representation. Generally, the wavelet basis is chosen such that the perturbation coefficients have desirable attributes such as direct correlation with some measure of error which is introduced at a given level of representation. During reconstruction from the wavelet representation, sufficiently small wavelet coefficients can be left out, resulting in a coarser approximation to the original data, with a known bound on the amount of error [34, 26, 39]. Further extensions have provided similar basis for the decomposition of surfaces [27]. Muraki [36] applies wavelets in 3D to compute multiresolution models of 3D volume data. Isosurfaces and planar cross sections of the resulting data show little change in image quality with large reductions in the amount of data representing the volume.

2.4 *Our Current Research*

In [37] we extend geometric reduction methods to arbitrary surfaces in 3D and to any number of data variables defined over the mesh by developing a algorithm for mapping from a surface mesh to a reduced representation and measuring the introduced error in both the geometry and the multivariate data. Furthermore, through error propagation, our

algorithm ensures that the errors in both the geometric representation and multivariate data do not exceed a user-specified upper bound. In addition, we describe a procedure by which a nested triangulation can be obtained, allowing smooth interpolation between meshes of varying resolution. The user-specified error bounds for geometry and data are intuitive, making it easy to guide or automate the mesh reduction process resulting in a mesh of desirable quality.

The algorithm for reduction follows the basic strategy of other "vertex deletion" schemes. From an initially dense surface mesh, vertices are considered as candidates for deletion. A candidate vertex is deleted if a valid retriangulation of the hole which results from deletion can be found. A valid retriangulation must maintain the topology of the original mesh, and the sum of the propagated errors and errors introduced from the deletion must be within user-specified bounds.

Unlike some mesh reduction schemes, there is no 'vertex classification' scheme which determines if a vertex is a candidate for deletion. All vertices are candidates, and may deleted so long as their removal does not violate the user-defined constraints. The error minimization scheme employed in retriangulation will automatically maintain edges in both the geometry and the data by choosing a retriangulation which is closest to the original data. Note that we consider only the cases of meshes which are 2-manifold, which are common in scientific data. For non-manifold meshes, classification may be required in order to determine if a vertex should be considered for deletion.

A desirable trait for a mesh reduction algorithm is the ability to create multiple nested levels of detail for an object, which can be smoothly interpolated and blended. Such methods are frequently used in real-time rendering systems such as a flight simulator, so that objects which are far away and appear small to the user may be rendered using a coarse resolution model [5]. As the object nears the user, the model can be smoothly blended to a more detailed resolution. Such methods would also be very useful in a navigable visualization environment. We describe a method to generate nested sets of models and interpolate between them.

The error control in the mesh reduction is driven by a mapping from one triangulation to another. Using this fact, we can develop a simple method for interpolating between multiple levels of triangulation. The first step is to generate multiple nested representations for a surface mesh. In a vertex removal mesh reduction, this is easily accomplished by generating meshes in the order of the most detailed to the most simplified mesh. At each stage of the mesh reduction, we use the previous mesh as a starting point and further reduce the mesh. We are guaranteed that the lower resolution meshes will contain only vertices from the higher resolutions.

In order to properly interpolate between the representations, we need only to store a description of the mapping which was used to create the reduced representations. When a retriangulation is projected to the original surface, the surface is segmented into linear

pieces. When interpolating between a higher and lower resolution model, it is these linear pieces which are interpolated from one resolution to the other. When the interpolation reaches the higher or lower resolution, the pieces can be replaced with the exact resolution model.

Figure 1 demonstrates the mesh reduction on a constant surface of density in a pion collision simulation. The surface was extracted through isosurfacing of a 70 x 40 x 25 volume of data. Seven variables, including density, pressure, and component velocities were all interpolated to the isosurface. Reduced meshes were computed with relative error bounds of 3% and 6%. Three pseudocolored surfaces are shown, representing three of the variables in the mesh. The reduced meshes are 70% - 80% smaller than the original mesh, while maintaining the geometry and data on the mesh.

In figure 2, an isosurface was extracted from a 54 x 24 x 24 volume mesh from a bullet impact simulation. Six variables defined on the mesh were interpolated to the surface. Meshes of reduced resolution were computed with relative error bounds of 3% and 9%. Three pseudocolored surfaces show the values of three of the variables on the original mesh as well as the simplified meshes. Mesh simplification reduced the size of the meshes by 50% - 70%.

3 Data Reconstruction

Reconstructing a topologically consistent and geometrically accurate model of the object and of its associated physical properties allows for the rapid display of different visualizations as well as supports interactive querying and manipulation. Several types of sensors are available to scan a 3D object and measure the location of points on its surface. Mechanical probes, used in the manufacturing industry, are extremely accurate but very slow to use. Laser Range Scanners can measure the location of a large number of points in a relatively small time. Recent models are also capable of capturing the RGB components of the color of the surface at each sample point. Less accurate (and cheaper) hand-held devices, based on ultrasound or magnetic fields, are also being marketed. Assuming no particular organization of the data allows us to treat several variations of the problem with a uniform approach, and constitutes an interesting theoretical challenge. The approach one might take in trying to solve this problem depends heavily on the assumptions that can be made about the sampled object and the data itself. For example, the object might be assumed to be smooth or instead contain creases and corners. The sampling might be very dense and uniform, or rather sparse.

3.1 *Piecewise-linear reconstruction*

In a short paper dated 1981, O'Rourke [18] explores the use of polyhedra to represent the "most reasonable" reconstruction of an object from a set of points. In particular, he proposes polyhedra of minimal surface area as a "natural" model for a given set of points, and describes an algorithm for the computation of an approximation of such polyhedra. The problem of reconstructing a *polygon* of minimal perimeter having a given set of points as vertices is easily proved to be equivalent to the Euclidean Traveling Salesman problem, which is known to be NP-hard. The problem of computing a polyhedron of minimal surface area is the 3D version of the minimal perimeter polygon, and is conjectured to be NP-hard as well.

Boissonnat [9] proposes two methods to build a triangulation having the given points as vertices. The first method is "local" and surface-based, whereas the second one is "global" and volume-based. Following his first approach, one starts with creating an edge between the two closest points. A third point is then chosen and added, so that a triangle is formed. Other points are successively added and new triangles are created, and joined to an edge of the current triangulation boundary, until all points have been included. The selection of which point to insert and to which edge to join the new triangle is based on a visibility heuristic. The second method proposed is based on the idea of first computing a Delaunay triangulation of the convex hull of the set of points, and then *sculpting* the volume by removing tetrahedra, until all points are on its boundary, or no tetrahedra can be further removed. The author suggests a heuristic to select which tetrahedron to remove.

Choi et al. [10] propose a method to grow a triangulation, starting from an initial triangle, based on the assumption that there exists a point from which all the points of the surface are visible. After a triangulation is built, it is improved by edge swapping based on a *smoothness* criterion. Veltkamp [24] introduces a new geometric structure, called γ-graph, which contains as a special case many well known geometric graphs, such as the Euclidean Minimum Spanning Tree, the Delaunay Triangulation, the Convex Hull and the Gabriel Graph. The γ-graph is progressively *constricted* (i.e. tetrahedra having boundary faces are deleted) until the boundary of the γ-graph is a closed surface, passing through all the given points.

Hoppe et al. [14] use an approach based on *normal propagation* to correctly orient the approximated manifold. Their method next constructs a regular subdivision of a parallelepiped containing the data points into cubes. The value of a function δ is computed at all vertices of the subdivision as the signed distance of the vertex from the oriented plane associated with the closest point in P. An algorithm similar to *marching cubes* is then used to construct a piecewise-linear approximation of the zero contour of δ. In two subsequent steps, described in [15, 13], the constructed mesh is optimized (i.e., the number of triangles is reduced while the distance of the mesh from the data points is kept small) and then a

smooth surface is built on it. While this approach gives very convincing results, and allows for smooth objects with sharp features to be correctly reconstructed, the computational time required by the optimization and smoothing steps is significant.

3.2 *Piecewise curved surface fitting*

While the theory for approximating a set of points, under certain constraints, with a polynomial of given degree is well developed, all these methods have to deal with the problem of inferring the right topology for the surface.

An early application of this method is described by Shmitt et al. [22]. The input points are assumed organized in a rectangular grid, and are adaptively fitted using Bernstein-Bézier parametric bicubic patches, joined to form a G^1 continuous surface. The approximation process begins with a rough approximating surface and uses subdivision to achieve the needed level of accuracy. The method is applicable to a restricted, yet important, class of objects, namely those for which a rectangular grid of points suffices to describe the surface (for example objects with an "almost cylindrical" geometry, like a human head). Objects of genus greater than 0, or objects with a convoluted geometry, require multiple point of views to be completely scanned.

Moore and Warren [17] describe a method for fitting *algebraic* surfaces to scattered dense data. Their method is adaptive and able in principle of dealing with complex geometry and topology. The fitting begins with a uniform mesh of tetrahedral elements that fill a region containing the data points. Then for each element in the mesh that contains points, a surface patch that approximates the points is computed based on least squares fit of the data points and of *auxiliary* data. It is known that least squares approximation of data points with algebraic patches might produce surfaces having extraneous parts. The auxiliary data serves the purpose of avoiding extraneous surface sheets. This data is an approximate sampling of the *signed-distance* function $\delta(x, y, z)$. They give examples of C^0 reconstruction of surfaces and briefly discuss a C^1 method (non adaptive) based on biquadratic, tensor-product implicit patches.

Goshtasby [12] introduces a new representation for parametric curves and surfaces, using Gaussian bases in rational form, and shows applications to the recovery of shape from noisy image data. The standard deviation of Gaussians control the smoothness of a recovered shape.

3.3 *Physically Based Modeling*

Another class of algorithms is based on the idea of *deforming* an initial approximation of a shape, under the effect of external forces and internal reactions and constraints. Terzopoulos et al. [23] use an elastically-deformable model with intrinsic forces that induce a preference for symmetric shapes, and apply them to the reconstruction of shapes from images. The

algorithm is also capable of inferring non-rigid motion of an object from time-varying images.

Pentland and Sclaroff [19] adopt an approach based on the finite element method and parametric surfaces. They start with a simple solid model (like a sphere or cylinder) and attach virtual "springs" between each data point and a point on the surface. The equilibrium condition of this dynamic system is the reconstructed shape. They show how the set of parameters that describe the recovered shape can be used in object recognition. Similar approaches are described in [21, 20, 16].

3.4 *Our Current Research*

Our research has focused on reconstruction algorithms [8, 7, 1] based on piecewise algebraic surface splines [2, 3]. Algebraic surface splines have many attractive qualities, from the closure properties with respect to important modeling operations such as intersection and blending, to a high design flexibility for a relatively low algebraic degree [6].

We consider the following reconstruction problem:

> Let an unorganized collection of points $P = \{(x_i, y_i, z_i)\} \subset \mathbf{R}^3$ and associated values $W = \{w_i\} \subset \mathbf{R}^1$, $i = 1 \ldots n$, be given. The points P are assumed to be sampled from a domain D in $\mathbf{R}^3$ (the boundary of a three-dimensional object) while the values W are assumed sampled from some scalar function F on the domain D.
>
> Construct a C^1 smooth piecewise polynomial surface $S^D : f^D(x, y, z) = 0$ and a C^1 smooth piecewise polynomial function (surface-on-surface) $S^F : f^F(x, y, z)$ on some domain that contains P such that, for $i = 1 \ldots n$:
>
> 1. $|f^D(x_i, y_i, z_i)| < \varepsilon^D$
> 2. $|f^F(x_i, y_i, z_i) - w_i| < \varepsilon^F$
>
> where ε^D and ε^F are user-defined approximation parameters. The user can also choose the degree of the Bernstein-Bézier polynomial patches used in the approximation.

Our approach aims at the following objectives:

1. **Unrestricted topological genus.** The method should be able to reconstruct objects of arbitrary genus.

2. **Approximation of the data.** Since the data set is noisy and usually very dense, attempting to interpolate all data points would lead to inefficiency and to an unnecessarily large number of patches.

3. **Adaptiveness.** Many objects have localized, small-scale features and large, flat or constant-curvature areas. Therefore it is convenient to be able to use patches of different size in different areas of the object surface.

4. **Tangent-plane continuity.** Often smooth manufactured objects can be modeled quite accurately with surface patches joining so that first-order derivatives across the common boundary are continuous. The present approach cannot automatically handle objects with *mixed* continuity (i.e., objects whose surface is formed by smooth regions that join at a crisp edge or corner). This will be the subject of future investigation.

5. **Reconstruction of a field over the surface.** In many important cases a scalar field has been sampled at the data points locations. The algorithm must be able to reconstruct a C^1-continuous function, defined over the object surface, that approximates the sampled data.

We have developed, implemented and experimented with several solutions to the reconstruction problem. Our algorithms consist of the following phases (for a detailed description of the reconstruction algorithms see [8, 7]):

1. **Build an approximation of the signed-distance function.** If M is a connected and orientable surface in $\mathbf{R}^3$, then it is possible to define (in all $\mathbf{R}^3$) a function $\delta(p)$, called signed-distance, by

$$\delta(p) := sign \cdot dist(p, M)$$

   where $dist(p, M)$ denotes the Hausdorff distance from the point p to the surface M and the $sign$ is chosen so that $\delta(p)$ is positive when p is on one side of M, and negative when p lies on the other side. Then $\delta(p) = 0$ will recover the surface M.

   When only discrete data on a surface is available, an approximate signed-distance can be defined in some appropriate way (see e.g. [17, 14, 8]). In particular, we have tried both the *normal propagation* approach proposed in [14], and a technique based on α-shapes [11], to define the signed-distance function (notice that this requires a topologically consistent reconstruction of the surface orientation at each point).

   This step can be seen as transforming the problem from a surface-data reconstruction to a volume-data approximation.

2. **Approximate the signed-distance by a piecewise polynomial function.** Build, in an adaptive fashion, a piecewise polynomial approximation $f^D(x, y, z)$ of $\delta(p, M)$. The piecewise polynomial is built by least squares fitting of trivariate polynomials, in each cell of a decomposition of a domain containing P, to the data points within the cell and to additional samples of the signed-distance function δ defined in phase 1 above. If the error-of-fit in a cell of the subdivision exceeds the given bounds, then

the decomposition is locally refined and the process is repeated in each new cell. The reconstructed domain is implicitly defined as $S^D(x, y, z) = 0$.

Both tensor-product (Figure 4) and barycentric (Figure 3), C^1 smooth piecewise implicit Bezier patches have been used for the data fitting.

3. **Approximate the scalar field defined over M.** Concurrently to the approximation of the signed-distance function, a piecewise polynomial approximation of the scalar field can be computed in a similar fashion by least squares fitting of the scalar field data in each cell of the decomposition.

4. **Make the reconstructed surface C^1-smooth.** Two smoothing techniques, namely the *free-form blending* [17] and the three-dimensional Clough-Tocher split [25] have been employed to achieve C^1-continuity.

The data for the human femur in Figure 3, 9223 points, comes from contouring of a CT scan. The algorithm does not use the fact that the data is arranged in slices. The reconstructed C^1 surface is made by 400 barycentric cubic, implicit Bezier patches. The reconstruction algorithm took about 6 minutes on a SGI MIPS4400 workstation.

The sampled points in Figure 4 come from the surface of a jet engine, while the associated values measure the pressure on the engine (data from a simulation). Figure 4 shows several steps of the reconstruction process on one part of the engine. The 3780 data points for the outer cowl are preprocessed to associate local fitting planes and orient the associated normals (4(a)). The approximation algorithm begins with a given grid (in this case, a uniform subdivision into $5 \times 5 \times 5$ equally-sized cubes) and then adaptively refines it until the error bound conditions are met (the error in this example was set to 0.01 times the max size of the object). The final subdivision is displayed in Figure 4(b). After averaging, a C^1-smooth piecewise polynomial surface is obtained, as shown in Figure 4(c). The complete reconstruction of this object took about 30 seconds on a SGI MIPS4400 workstation. The full reconstructed engine is finally shown in Figure 4(d). At the same time, a different piecewise polynomial, whose domain is the same as for the surface implicit function, approximates the sampled scalar field (see Figure 5). Figure 5 shows four different visualization of the jet engine data. In Figure 5(a) some isoregions of the pressure field have been drawn on the engine surface. In Figure 5(b) we have used the normal-projection method to show the scalar field as a surface-on-surface: Points on the domain surface have been projected along the surface normal direction to a distance proportional to the value of the field at that point. The field surface patches are visible in Figure 5(c). Finally, in Figure 5(d), we show isocontours projected on the surface-on-surface.

4 Acknowledgments

This survey is abstracted from papers written in collaboration with Fausto Bernardini, Daniel Schikore, and Guoliang Xu. I am grateful to the Lawrence Livermore National Lab for access to the pion collision and bullet impact data sets, the Lafayette Orthopedic Clinic for the femur data, and NASA for the jet engine data set. I also thank Myung-Soo Kim for his careful reading of the manuscript. This research has been supported in part by NSF grant CCR 92-22467, AFOSR grant F49620-94-1-0080, ONR grant N00014-94-1-0370 and ARO grant DAAH04-95-1-0008.

References

[1] C. Bajaj and G. Xu. Modeling Scattered Function Data on Curved Surface. In J. Chen, N. Thalmann, Z. Tang, and D. Thalmann, editor, *Fundamentals of Computer Graphics*, pages 19 – 29, Beijing, China, 1994.

[2] C. Bajaj and I. Ihm. Smoothing Polyhedra with Implicit Algebraic Splines. *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 19–26, August 1992.

[3] C. Bajaj, J. Chen, and G. Xu. Modeling with Cubic A-Patches. *ACM Transactions on Graphics*, April 1995.

[4] J. Lee. Comparison of existing methods for building triangular irregular networks. *Int. Journal of Geographical Information Systems*, 5(2):267–285, 1991.

[5] J. Rohlf and J. Helman. IRIS performer: A high performance multiprocessing toolkit for real–Time 3D graphics. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 381–395. ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0.

[6] C. Bajaj. The emergence of algebraic curves and surfaces in geometric design. In *Directions in Geometric Computing*, R. Martin, Ed. Information Geometers Press, 1993, pp. 1–29.

[7] C. Bajaj, F. Bernardini, and G. Xu. Adaptive reconstruction of surfaces and scalar fields from dense scattered trivariate data. Computer Science Technical Report CSD-TR-95-028, Purdue University, 1995.

[8] C. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *Computer Graphics Proceedings* (1995), Annual Conference Series. Proceedings of SIGGRAPH 95, ACM SIGGRAPH. To appear.

[9] J. Boissonat. Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics 3*, 4 (Oct. 1984), 266–286.

[10] B.K. Choi, H.Y. Shin, Y.I. Yoon, and J.W. Lee. Triangulation of scattered data in 3D space. *Computer Aided Design 20*, 5 (June 1988), 239–248.

[11] H. Edelsbrunner and E. Mücke. Three-dimensional alpha shapes. *ACM Trans. Graph. 13*, 1 (Jan. 1994), 43–72.

[12] A. Goshtasby. Design and recovery of 2-D and 3-D shapes using rational Gaussian curves and surfaces. *International Journal of Computer Vision 10*, 3 (1993), 233–256.

[13] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schwitzer, and W. Stuelzle. Piecewise smooth surface reconstruction. In *Computer Graphics Proceedings* (1994), Annual Conference Series. Proceedings of SIGGRAPH 94, ACM SIGGRAPH, pp. 295–302.

[14] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuelzle. Surface reconstruction from unorganized points. *Computer Graphics 26*, 2 (July 1992), 71–78. Proceedings of SIGGRAPH 92.

[15] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuelzle. Mesh optimization. In *Computer Graphics Proceedings* (1993), Annual Conference Series. Proceedings of SIGGRAPH 93, ACM SIGGRAPH, pp. 19–26.

[16] C. Liao, and G. Medioni. Surface approximation of a cloud of 3D points. *Graphical Models and Image Processing 57*, 1 (Jan. 1995), 67–74.

[17] D. Moore and J. Warren. Approximation of dense scattered data using algebraic surfaces. In *Proceedings of the 24th annual Hawaii International Conference on System Sciences* (1991), V. Milutinovic and B. D. Shriver, Eds., vol. 1.

[18] J. O'Rourke. Polyhedra of minimal area as 3D object models. In *Proc. of the International Joint Conference on Artificial Intelligence* (1981), pp. 664–666.

[19] A. Pentland and S. Sclaroff. Closed form solutions for physically based shape modelling and recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 7 (1991), 715–729.

[20] R. Poli, G. Coppini, and G. Valli. Recovery of 3D closed surfaces from sparse data. *Computer Vision, Graphics and Image Processing 60*, 1 (July 1994), 1–25.

[21] D. Ruprecht and H. Muller. Free form deformation with scattered data interpolation methods. In *Geometric Modelling (Computing Suppl. 8)*, H. H. G, Farin and H. Noltemeier, Eds. Springer-Verlag, Wien, 1993, pp. 267–281.

[22] F. Schmitt, B. Barsky, and W. Du. An adaptive subdivision method for surface fitting from sampled data. *Computer Graphics 20*, 4 (1986), 179–188. Proceedings of SIGGRAPH 86.

[23] D. Terzopoulos, A. Witkin, and M. Kass. Constraints on deformable models: recovering 3D shape and non-rigid motion. *Artificial Intelligence 36* (1988), 91–123.

[24] R. Veltkamp. *Closed object boundaries from scattered points*. PhD thesis, Center for Mathematics and Computer Science, Amsterdam, 1992.

[25] A. Worsey and G. Farin. An n-dimensional clough-tocher interpolant. *Constructive Approximation 3*, 2 (1987), 99–110.

[26] R. DeVore, B. Jawerth, and B. Lucier. Image compression through wavelet transform coding. *IEEE Transactions on Information Theory*, 38:719–746, 1992.

[27] R. DeVore, B. Jawerth, and B. Lucier. Surface compression. *Computer Aided Geometric Design*, 9:219–239, 1992.

[28] R. DeVore and B. Lucier. Wavelets. In A. Iserles, editor, *Acta Numerica 92*, pages 1–56. Cambridge University Press, 1992.

[29] R. Fowler and J. Little. Automatic extraction of irregular network digital terrain models. In *Computer Graphics (SIGGRAPH '79 Proceedings)*, volume 13(3), pages 199–207, August 1979.

[30] T. Funkhouser and C. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 247–254, August 1993.

[31] P. Hinker and C. Hansen. Geometric optimization. In Gregory M. Nielson and Daniel Bergeron, editors, *Proceedings of Visualization '93 (San Jose, California, October 25–29, 1993)*, pages 189–195. IEEE Computer Society, IEEE Computer Society Press, October 1993. ISBN 0-8186-3940-7.

[32] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 19–26, August 1993.

[33] W. Lorensen and H. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 163–169, July 1987.

[34] B. Lucier. Wavelets and image compression. In T. Lyche and L. Schumaker, editors, *Mathematical Methods in Computer Aided Geometric Design II*, pages 391–400. Academic Press, 1992.

[35] S. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, July 1989.

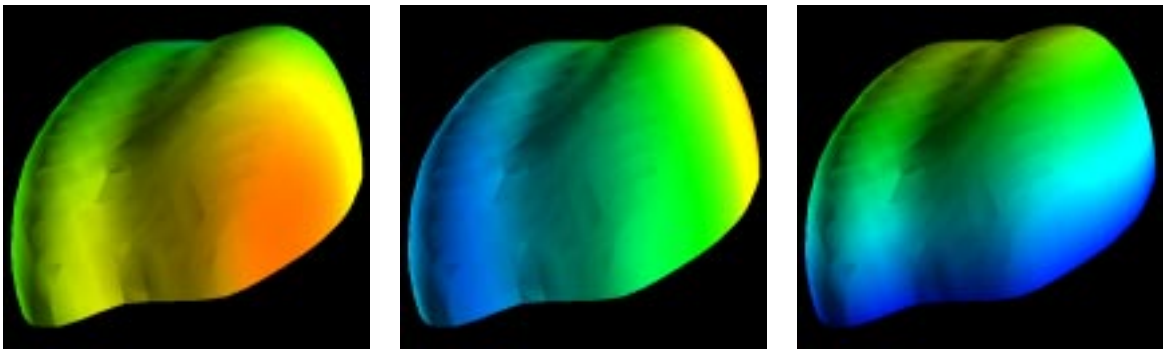
[36] S. Muraki. Approximation and rendering of volume data using wavelet transforms. In Arie E. Kaufman and Gregory M. Nielson, editors, *Proceedings of Visualization '92 (Boston, Massachusetts, October 19–23, 1992)*, pages 21–28. IEEE Computer Society, IEEE Computer Society Press, October 1992.

[37] C. Bajaj and D. Schikore. Error-bounded Reduction of Triangle Meshes with Multivariate Data. Technical Report CS-95-034, Purdue University, Computer Science, February 1995.

[38] W. Schroeder, J. Zarge, and W. Lorensen. Decimation of triangle meshes. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26(2), pages 65–70, July 1992.

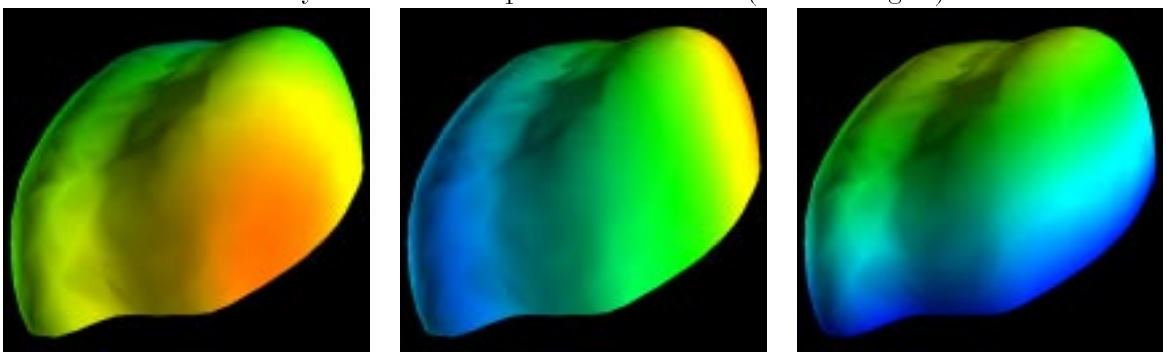
[39] E. Stollnitz, T. DeRose, and D. Salesin. Wavelets for computer graphics: A primer. University of Washington, 1994.

[40] G. Turk. Re-tiling polygonal surfaces. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26(2), pages 55–64, July 1992.



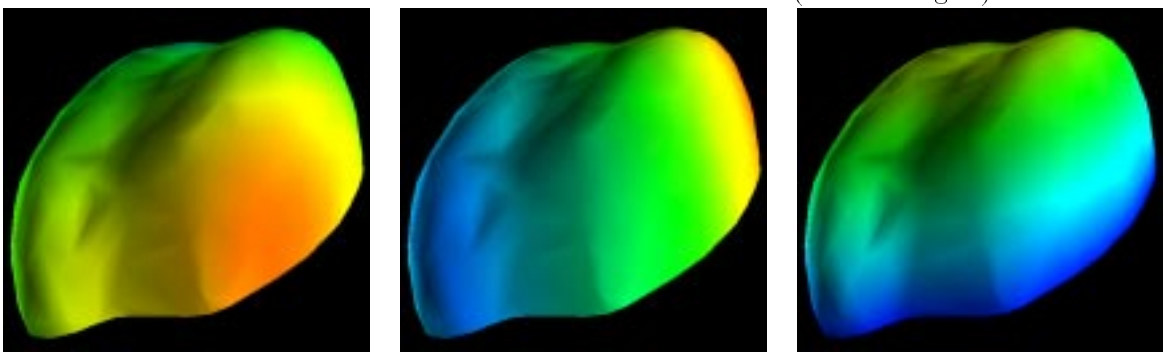
(a) (b) (c)

Density surfaces from pion collision data (3450 triangles)



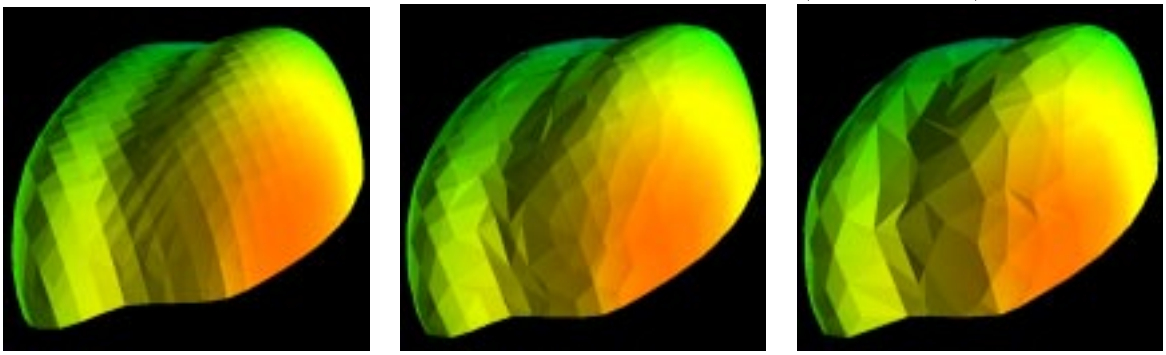
(d) (e) (f)

Reduced surfaces with 3% relative error bounds (1043 triangles)



(g) (h) (i)

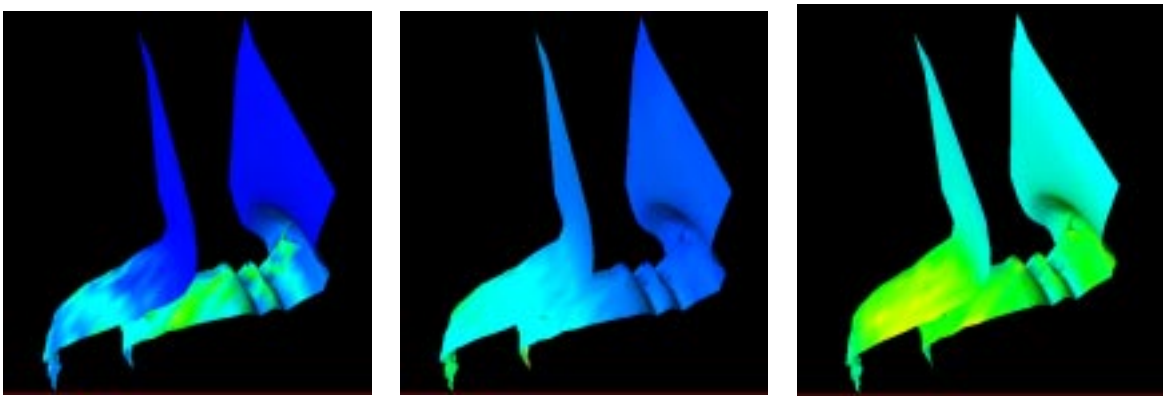
Reduced surfaces with 6% relative error bounds (660 triangles)



(j) (k) (l)

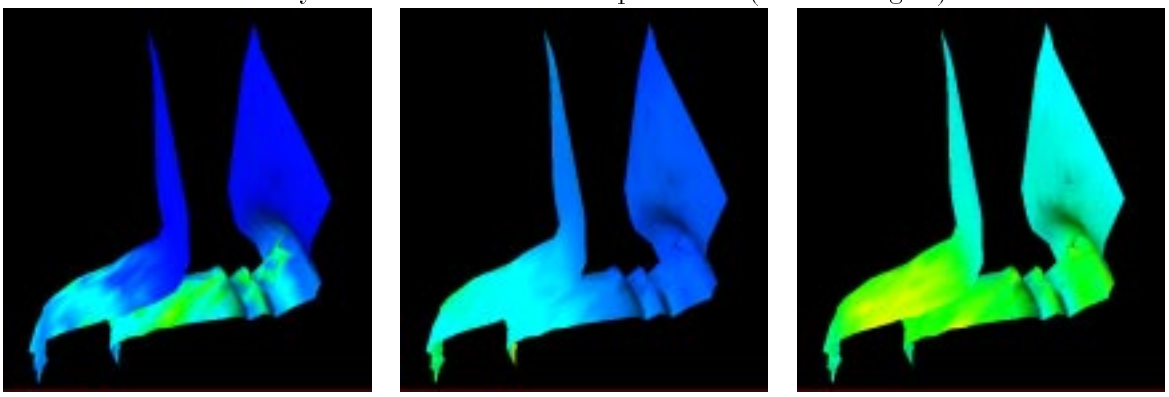
Flat shaded surfaces of (a), (d), and (g) reveal the tesselation

Figure 1: Original surface data with two levels of mesh reduction, shown for 3 variables



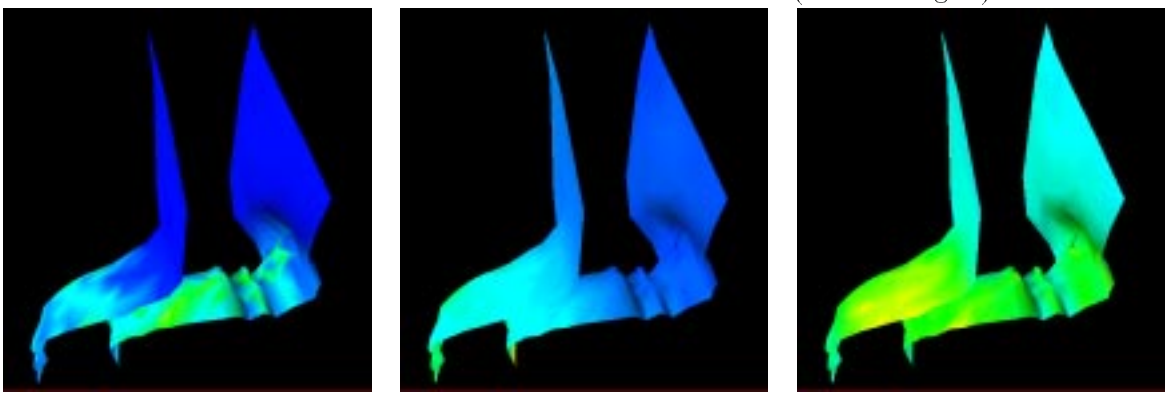
(a) (b) (c)

Density surfaces from bullet impact data (2869 triangles)



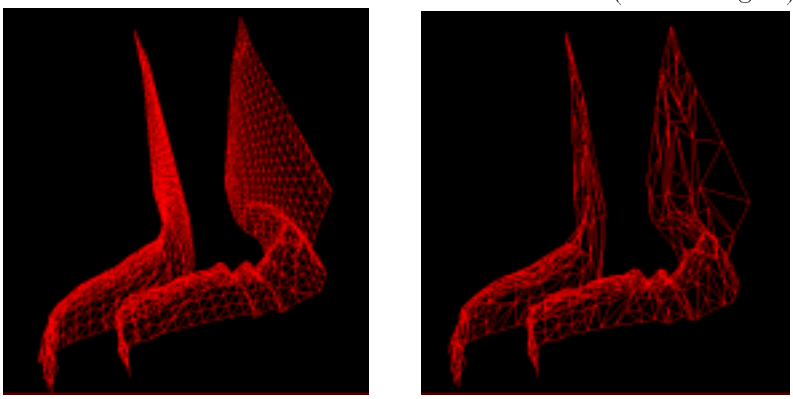
(d) (e) (f)

Reduced surfaces with 3% relative error bounds (1356 triangles)



(g) (h) (i)

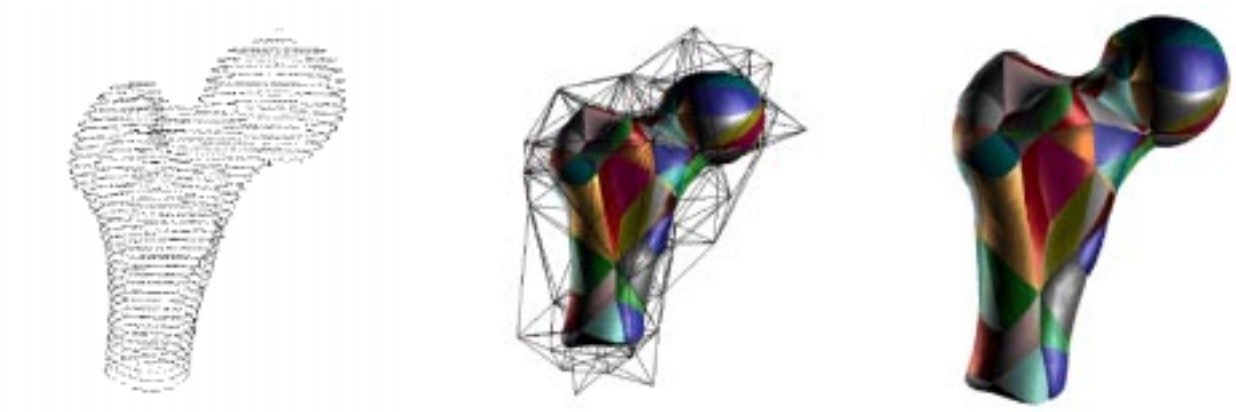
Reduced surfaces with 9% relative error bounds (844 triangles)



(j) (k)

Wireframe surfaces of (a-c) and (g-i)

Figure 2: Original surface data with two levels of mesh reduction, shown for 3 variables



(a) (b) (c)

Figure 3: (a) Data set for the upper part of a human femur. Data from a CT scan. (b) Final decomposition (wireframe). (c) Reconstructed object.

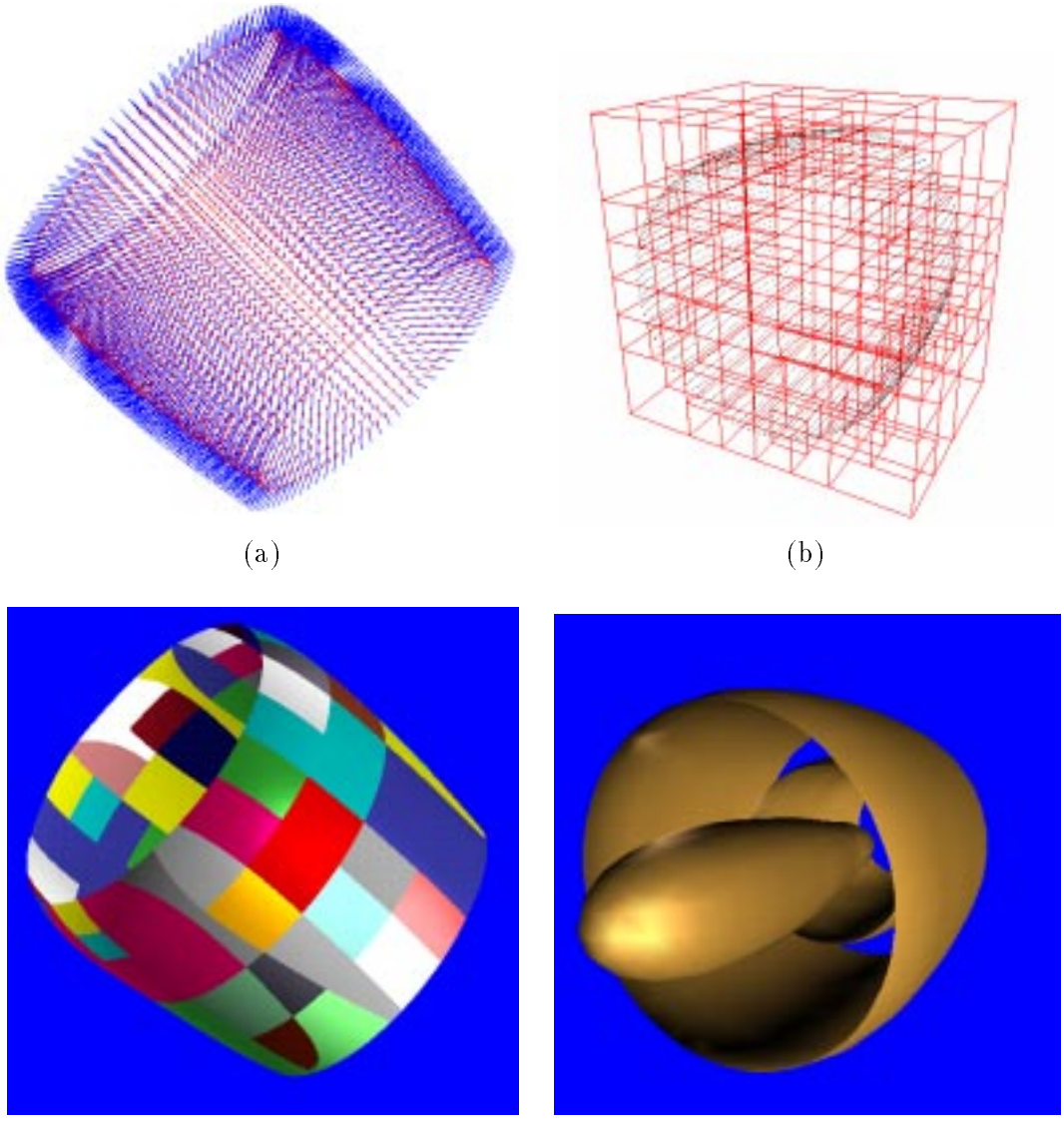


Figure 4: Reconstruction of a jet engine: (a) Input data for the outer cowl, with the oriented normals. (b) Octree subdivision generated by the approximation algorithm. (c) Piecewise polynomial approximation. (d) Reconstructed engine.

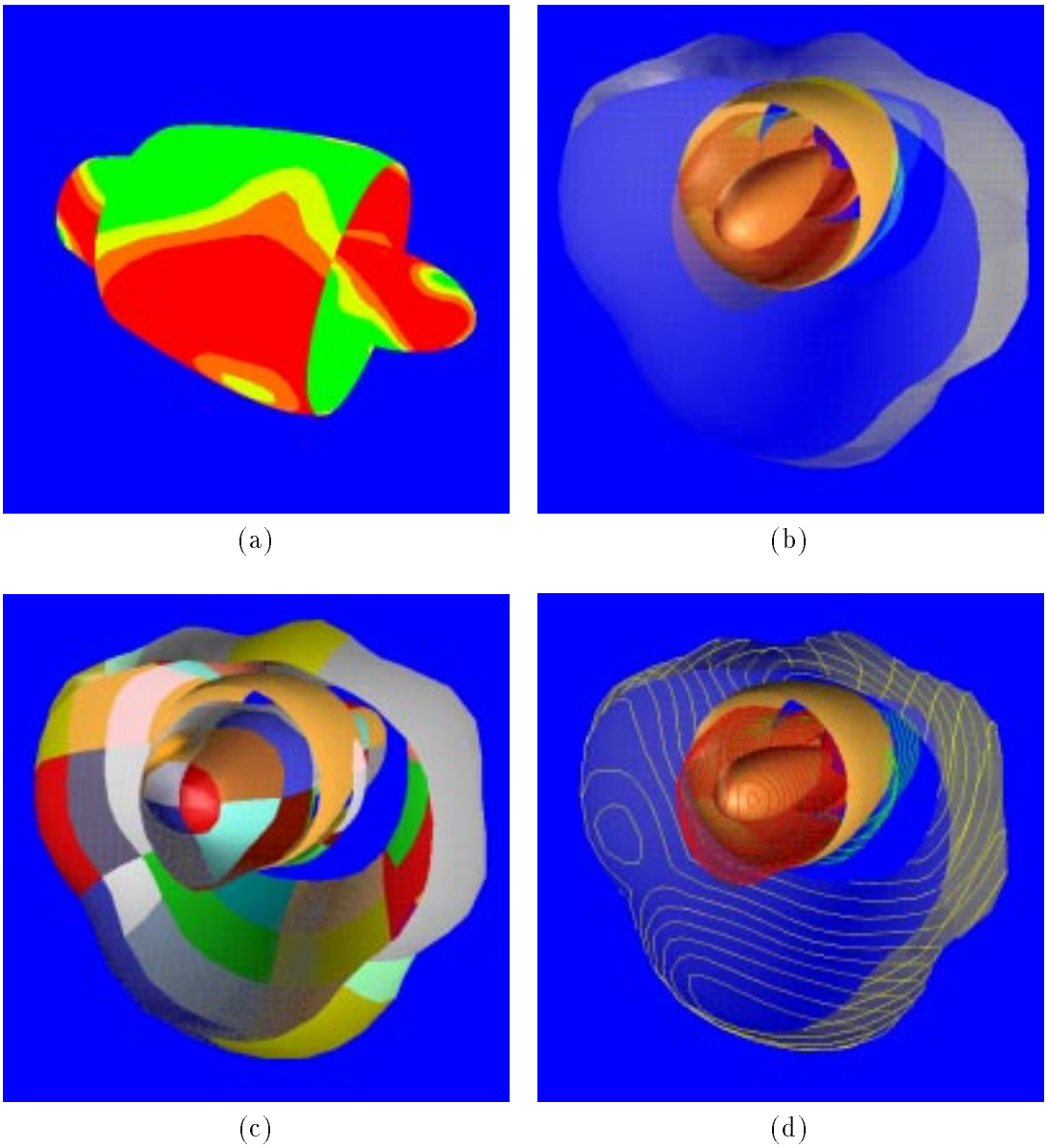


Figure 5: Visualization of the reconstructed jet engine and a pressure field on its surface: (a) Iso regions of the pressure field. (b) The pressure field displayed with the normal projection method (surface on surface). (c) The piecewise polynomial patches. (d) Isocontours of the pressure field displayed on the projected surface.