Re-Tiling Polygonal Surfaces

Greg Turk
Department of Computer Science
University of North Carolina at Chapel Hill

Abstract

This paper presents an automatic method of creating surface models at several levels of detail from an original polygonal description of a given object. Representing models at various levels of detail is important for achieving high frame rates in interactive graphics applications and also for speeding-up the off-line rendering of complex scenes. Unfortunately, generating these levels of detail is a time-consuming task usually left to a human modeler. This paper shows how a new set of vertices can be distributed over the surface of a model and connected to one another to create a re-tiling of a surface that is faithful to both the geometry and the topology of the original surface. The main contributions of this paper are: 1) a robust method of connecting together new vertices over a surface, 2) a way of using an estimate of surface curvature to distribute more new vertices at regions of higher curvature and 3) a method of smoothly interpolating between models that represent the same object at different levels of detail. The key notion in the re-tiling procedure is the creation of an intermediate model called the *mutual tessellation* of a surface that contains both the vertices from the original model and the new points that are to become vertices in the re-tiled surface. The new model is then created by removing each original vertex and locally re-triangulating the surface in a way that matches the local connectedness of the initial surface. This technique for surface re-tessellation has been successfully applied to iso-surface models derived from volume data, Connolly surface molecular models and a tessellation of a minimal surface of interest to mathematicians.

CR Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation – Display algorithms; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modelling – Curve, surface, solid, and object representations.

Additional Key Words and Phrases: model simplification, automatic mesh generation, constrained triangulation, levels-of-detail, shape interpolation.

1 Introduction

This paper shows how a simplified polygonal model can be automatically created from an initial polygonal description of an object. We use the term *re-tiling* to describe the process of simplifying a polygonal model. The notion of representing a model at multiple levels of detail is a common thread that runs through much work in computer graphics and image processing. These levels of detail can be found in a number of forms, such as multiple collections of polygons, different collections of bicubic surface patches or variously filtered levels of a raster image. There are several benefits to having more than one representation of an object. One benefit is that it is often unnecessary to use a fully-detailed model of an object during rendering if the object will cover a small portion of the screen. Using a smaller model can significantly shorten the time it takes to render an image. It is this ability to increase the rendering rate, especially for interactive applications, that motivates the work presented in this paper. Another benefit of having more than one representation of an object is that this is often a graceful way to avoid sampling problems when rendering an image. Probably the best-known example of this in computer graphics is the texture anti-aliasing work of Lance Williams [Williams 83]. A third reason for using multiple levels of detail is that features of an object can be classified by following the features through successively more coarse representations of the object. This method of feature recognition appears in much of the recent work being done in image processing and pattern recognition. Computer graphics has yet to make much use of feature tracking and elimination, and we will return to this issue in the future work section of this paper.

Polygonal descriptions of objects are currently the most widely-used forms of model representation in computer graphics. One reason for this is the availability of graphics workstations that can rapidly render polygons. Another reason is that there are a large numbers of techniques for translating a given model into a polygonal dataset. For these and other reasons, it is likely that polygonal representations of objects will continue to be important to computer graphics. This serves as motivation for finding automatic methods of creating new polygonal models of the same object that have a fewer number of polygons than the original description.

Because there is such a wide range of objects that can be represented by polygonal tessellations, it may be impossible to find one technique that can do a good job of re-tiling any given polygon dataset. For instance, techniques that are successful at reducing the number of polygons in a model of a building may not necessarily be applicable to re-tiling of medical datasets such as those derived from CT scans. This paper's re-tiling method is best suited to models that represent curved surfaces. Examples of such models include iso-surfaces from medical data and from molecular graphics, smooth mathematically-defined manifolds and digitized or hand-modelled organic forms such as animals or people. This technique is poorly suited to models that have well-defined corners and sharp edges such as buildings, furniture and machine parts.

This paper begins with an overview of related work in creating levels of detail, and, in particular, work that deals with polygonal models. The rest of the paper describes the basic steps taken to re-tile a given model and several extensions to this basic method. The first section on re-tiling describes how to distribute a given number of points evenly over a polygonal surface. These points will eventually become the vertices of the new model. Next, the notion of *mutual tessellation* followed by *vertex removal* is presented as a robust method of completely replacing the original set of vertices with the new points. This is how a completely new triangulation of the model is created. The next section shows how local estimates of maximum curvature can be used to concentrate more new vertices at regions that need more points to faithfully represent the surface. The paper then describes how the polygons from a more fine representation of an object can be flattened onto the surface of a more coarse polygonal model. Using this method, we can interpolate between this flattened version of the model and the original high-detail representation to give a smooth transition between the coarse and the fine versions of an object. The next section describes how re-tiled models can aid the interactive task of radiation treatment planning. The final section discusses future topics of research in representing multiple levels of detail in polygonal models.

2 Previous Work

James Clark's paper on hierarchical geometric models describes the benefits of using more than one representation of a model for image rendering [Clark 76]. Clark points out that objects that cover a small area of the screen can be rendered from a simplified version of the object and that this allows more efficient rendering of a scene. This same benefit of having both simple and complex representations of an object is given by Frank Crow in his paper on an image generation environment [Crow 82]. Crow gives the example of a chair that is represented in high detail, medium detail and very low detail. The three models in his example were created by hand, but Crow suggests that creating the lower levels of detail is a process that should be automated. A guaranteed frame-rate is essential in flight simulators, and for this reason models of objects such as airplanes are often made at several levels of detail by hand [Cosman & Schumacker 81].

The creation of lower levels of detail has been automated for some well-behaved polygonal datasets. Lance Williams showed how a regular mesh of quadrilaterals can be used to represent surfaces such as a human face, and how such meshes can be filtered down to smaller resolutions in the same manner as he used for texture filtering [Williams 83]. This is similar to how flight simulators use coarse versions of terrain data when a ground feature is far away and use a more detailed terrain model when the feature is closer to the viewer. The flight simulator literature describes how new features of the terrain can be gradually introduced as the viewer moves closer by first adding new vertices in the plane of a terrain polygon and then moving each vertex's elevation smoothly until it reaches the correct elevation [Zimmerman 87]. With a gridded terrain model it is easy to know which vertices need to be joined to form new polygons when a new vertex is added or when an old vertex is removed. This problem is more difficult for polygonal models with arbitrary topology.

Another polygonal data format that has been automatically re-tiled is the laser-scanned data from Cyberware Laboratories of Monterey, California. Their digitizing method results in a large collection of regularly joined quadrilaterals. Schmitt and co-workers have adaptively fit bicubic patches to such models by starting with a rough approximation of the surface and then adaptively refining the surface at locations where the model is not yet well fit [Schmitt 86]. This method generates models at varying levels of detail by specifying a set of increasingly fine tolerance levels for the surface fit. Extensions to this method have been explored to adapt the technique to creating polygonal models and to more closely bound the error [DeHaemer & Zyda 91]. DeHaemer and Zyda's methods reduced a 112,128 polygon image of a human head to 12,821 polygons. As with terrain data, the Cyberware format makes it easy to decide which vertices become neighbors when a vertex is added or removed.

To reduce the large numbers of polygons often found in medical data, Kevin Novins implemented a method of identifying and removing vertices that are in relatively flat portions of a polygonal object [Novins 92]. His program examines the variance in surface normals of triangles that share a given vertex and uses this to decide which vertices to remove. When a vertex is removed, the region immediately surrounding the vertex is re-triangulated. The user gives a target number of vertices and the program removes vertices until this number is reached. Schroeder and his co-workers have also used an approach of vertex removal and local re-triangulation for simplifying polygonal models [Schroeder et al 92]. They remove vertices that are within a distance tolerance of a plane that approximates the surface near the vertex. Their method also identifies sharp edges and sharp corners and makes sure that such features are retained in order to better represent the original data. They show how these techniques can be used to drastically reduce the number of polygons in large medical and terrain models and still retain feature detail.

There is a large body of literature on automatic mesh generation for use in finite element techniques. An overview of this work is given in [Ho-Le 88]. Here the problem is how to sub-divide the surface or volume of an object to provide a mesh over which some physical properties of the material can be simulated, such as heat dissipation or stress and strain. It is assumed that all the edges and faces of a model are to be accurately reflected in the re-meshed version of the object. This is the main difference between meshes used in finite element methods versus smooth surface models for rendering in computer graphics. The exact placement of vertices and edges in a polygonal representation of a cat are not as important as the placement of the edges separating the copper and iron portions of a machine part being analyzed for heat conductivity. There are some issues, however, that do touch upon the problems that are found in mesh generation in both domains. For example, many finite element meshing routines use local re-meshing operators to improve the shapes of triangles in an initial mesh. Similar local operations can be used to improve re-tilings for computer graphics.

There is a good deal of material in computational geometry that is relevant to the re-tiling problem. Specifically, the properties of Voronoi regions and the associated Delaunay triangulation are relevant to the question of "goodness" of triangle shape in a triangulation of a collection of points [De Floriani et al 85].

3 Choosing New Vertices for Re-Tiling

3.1 Input Surface and the Results of Re-Tiling

The re-tiling method described in this paper begins with a polygonal surface and creates a triangulation of this surface with a user-specified number of vertices. There are few restrictions on the initial polygonal surface. The polygons may be either concave or convex, and may in fact have holes. The major restriction is on the number of polygons that share any given edge. The method described below is suitable for polygonal models in which each edge is shared by either one or two polygons. If a model satisfies this restriction, the algorithm is guaranteed to produce a new model with the same topology as the original model. The method will not introduce tears

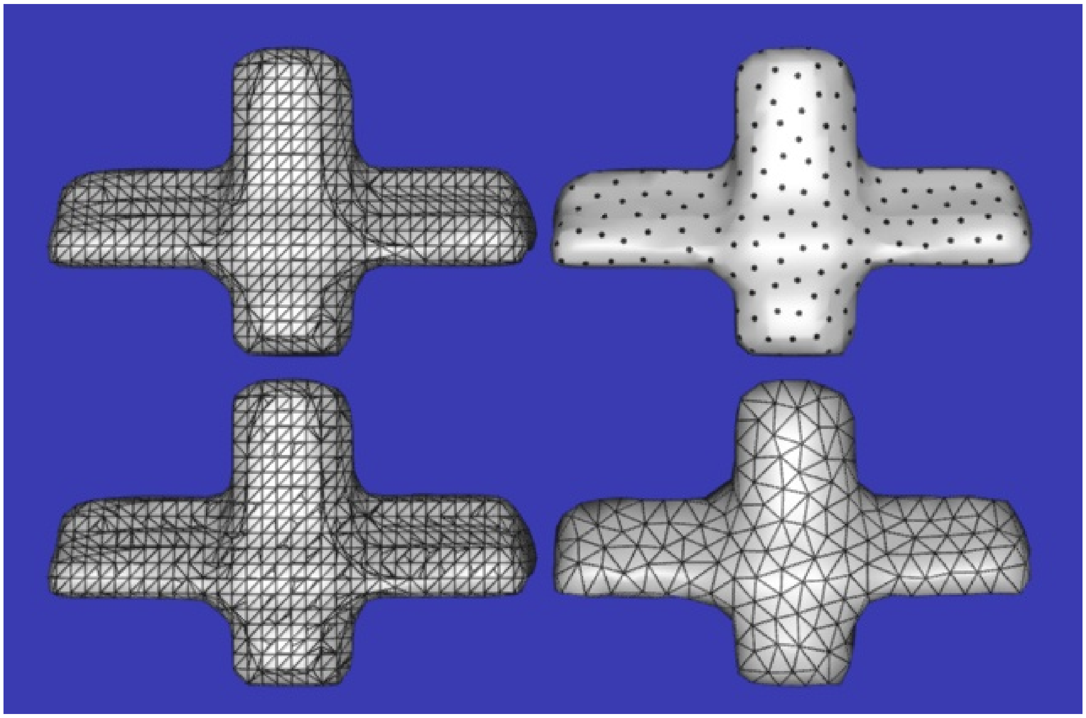


Figure 1: Re-tiling of a radiation iso-dose surface. Upper left: Original surface. Upper right: Candidate vertices after point-repulsion. Lower left: Mutual tessellation. Lower right: Final tessellation.

in the surface and will not connect regions of the surface that were unconnected in the original model. The next two sections outline the basic re-tiling method.

3.2 Positioning Vertices by Point Repulsion

The first step in re-tiling is to choose a set of points that will, at a later step, become the vertices of a new triangular tessellation of the surface. These new points are chosen to lie in the planes of the original polygons, and some of them may in fact be coincident with the vertices of the original model. The underlying assumption of this re-tiling approach is that the original polygonal surface gives a good indication of the location of the surface to be represented, but that the original placement of vertices on this surface may be poor choices for vertex positions of a re-tiled version of the surface. Allowing the new vertices to be placed anywhere on the surface lets them be placed in a manner that will give well-shaped triangles in the new representation of the surface. Placing the points fairly uniformly over the surface, as described in this section, is the portion of the re-tiling responsible for faithfully representing the *geometry*, the location and curvature, of the re-tiled surface. Joining these points together to form a triangular mesh, described in the following section, is that part of the re-tiling method responsible for faithfully representing the original surface's *topology*, that is, which parts of the surface connect to which other parts of the surface.

The basic method of placing these new points on the surface is taken directly from work on mesh generation for texture synthesis [Turk 91], and is described briefly below. This method places points uniformly over a given polygonal surface by distributing points at random over the surface and then having each point repel all of its neighbors. Sometimes, however, it is not desirable to have the distribution of points be uniform over the surface. This subject is addressed in a later section of this paper.

The re-tiling begins by having a given number of points (specified by the user) placed randomly over the surface of the polygonal model. Each point is placed by first making a random, area-weighted choice from among all the polygons in the model and placing the point at a random position on this polygon. Once all the points have been randomly placed on the surface, a relaxation procedure is applied to move each point away from all other nearby points. The basic operation of this relaxation procedure is to fold or project nearby points onto a plane tangent to the surface at one point, to calculate the repelling force that each nearby point has on the given point and then to move this point over the polygonal surface based on the force exerted against it. A point that is pushed off one polygon is moved onto an adjacent polygon. For the sake of speed, the repelling force that one point has on another is a force that falls off linearly with distance, and thus becomes zero at a fixed radius. Because points farther apart than this distance do not affect one another, the search for nearby points can be made constant-time by placing all of the points in a three-dimensional grid data structure. The upper right portion of Figure 1 shows 400 points that have been positioned on a polygonal surface by this relaxation procedure. The original model, with its polygons outlined in black, is shown in the upper left of the same figure. This model is a tessellation of a radiation dose level surface that has been used to help visualize radiation treatment beams. The original model contains 1513 vertices.

4 Re-Tiling by Mutual Tessellation

4.1 Some Pitfalls of Re-Tiling

Once the points that will become new vertices (the *candidate vertices*) have been placed on the model's surface, the next task is to find how these vertices can be connected together to form a triangular mesh that reflects the topology of the original surface. This is a difficult task because of the many pitfalls that a complicated surface can present. The need for a robust algorithm cannot be overly stressed. One problem case in connecting the candidate vertices is when two portions of a surface that are far from one another as measured over the surface are actually near to each other in 3-space because the surface folds back on itself (see Figure 2a). Any algorithm for connecting together the candidate vertices must not

Figure 2: Problems encountered when connecting new vertices. (a) Connecting regions that fold back near one another. (b) "Bubbles" resulting from incorrect joining of vertices.

join together a pair of vertices that reside on two such separated regions (the thin lines in Figure 2a). Another pitfall is the creation of small surface "bubbles", where two sets of polygons are created that both tile the same portion of a surface (see Figure 2b). Groups of polygons that meet at a sharp corner also present difficulties when re-tiling.

Two distinct approaches were tried for connecting the candidate vertices before the method described below (mutual tessellation) was found. Both of these earlier methods failed because they relied on heuristics to choose which candidate vertices were neighbors and which of these neighbors should be connected together to form triangles. The first of these failed methods used a planar approximation to a point's Voronoi region to determine neighbors. The second failed technique used a global greedy algorithm. This method added a new edge to the list of edges in the re-tiling if it was the shortest edge not already on the list and if it did not intersect any other edge in planar approximation to the surface in the area near the edge. Below we will see how a *local* greedy algorithm is used to give a robust re-tiling method.

4.2 Mutual Tessellation

The key notion in creating a re-tiling of the surface is to form an intermediate polygonal surface, called a *mutual tessellation*, that incorporates both the old vertices of the original surface and the new points that are to become vertices in the re-tiled surface. After the mutual tessellation is made, the old vertices are removed one at a time and the surface is re-tiled locally in a manner such that the new triangles accurately reflect the connectedness of the original surface. Creating the mutual tessellation is a straightforward task. Each polygon of the original model is replaced by a collection of triangles that exactly tiles the polygon but that also incorporates the candidate vertices that lie in the polygon. This re-triangulation of a given polygon is performed by first gathering together the vertices of the polygon along with the candidate vertices that lie on this polygon. This collection of points (original vertices and candidate vertices) is then triangulated, subject to the constraint that the edges of the original polygon are to be included in the final triangulation. The triangulation is performed in the plane of the given polygon. For example, a square polygon containing exactly one candidate vertex would be removed from the model and replaced by a set of four triangles that all meet at the one candidate vertex. If the square face contained n candidate vertices then it would be replaced by a set of $2n$+2 triangles.

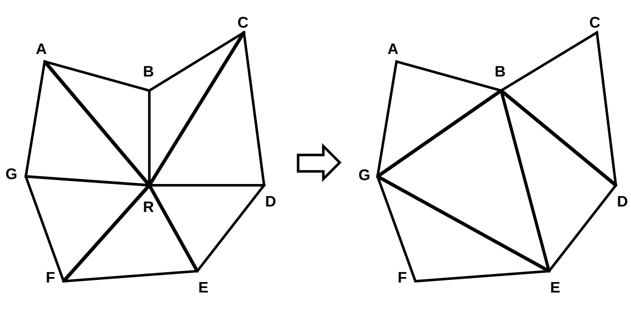


Figure 3: Removing a vertex from a mutual tessellation.

The method of constrained triangulation used for this paper is greedy triangulation, but this is just one of several ways to form such a triangulation [Preparata & Shamos 85]. There is no chance for misrepresenting the original surface at this stage because each polygon is replaced by a set of triangles that exactly tile the original polygon. An added benefit to using mutual tessellation is that the original surface can include concave polygons or even polygons with holes since constrained triangulation algorithms easily handle these cases.

The lower left object of Figure 1 shows the mutual tessellation of the original model shown in the upper left portion of the figure. The candidate points that have been used to create this tessellation are those shown in the upper right of the same figure.

4.3 Removing Old Vertices

The next task is to remove the old vertices in a way that guarantees that the newly-created triangles follow the topology of the original surface. This can be done by invoking the same triangulation routine that was used to create the mutual tessellation. Given an old vertex R to be removed, we collect together all vertices that share a triangle with R. Call this collection of neighboring vertices V, and give the name T to the set of triangles that the vertices in V share with the vertex R. Then this collection of neighboring vertices, *without* the vertex R, are projected onto a plane that is tangent to the surface at R. Now a few tests are made to see if this region can be re-tiled without compromising the topology of the surface. These tests are described later. If the tests check out, then the vertices V are triangulated along with the additional constraints that all edges of the triangles in T that do not contain R must be included in the final triangulation. Call these additional constraint edges the set E. This set of edges E form a closed polygon surrounding the vertex R. The triangulation is performed along with the final constraint that no new edges are to be introduced outside of the polygon formed by the edges in E.

Figure 3a shows a vertex R to be removed and its set V of neighbors: A, B, C, D, E, F and G. In this example there are seven triangles in the set T, and the set E consists of the edges AB, BC, CD, DE, EF, FG and GA. Figure 3b shows the result of removing R and triangulating the neighbors in V to give five new triangles. These new triangles completely replace the triangles in the set T, and all the triangles in T are removed from the model. Notice that performing this triangulation in a plane assures us that the new triangles match the topology of that local portion of the original surface. The newly-created triangles are constrained to have a common border that is just the edges in E, so they will be adjacent to the same triangles that used to border the triangles in T.

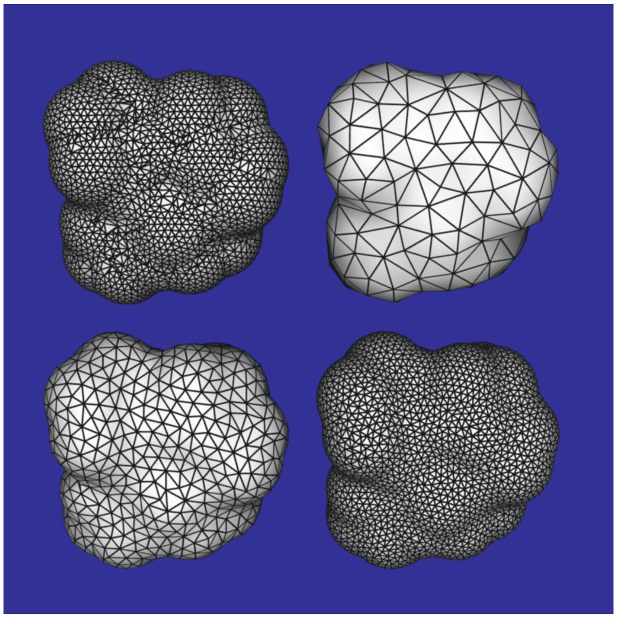


Figure 4: Re-tiling a molecular model. Original model is shown in the upper left. Other models are re-tilings of the same object.

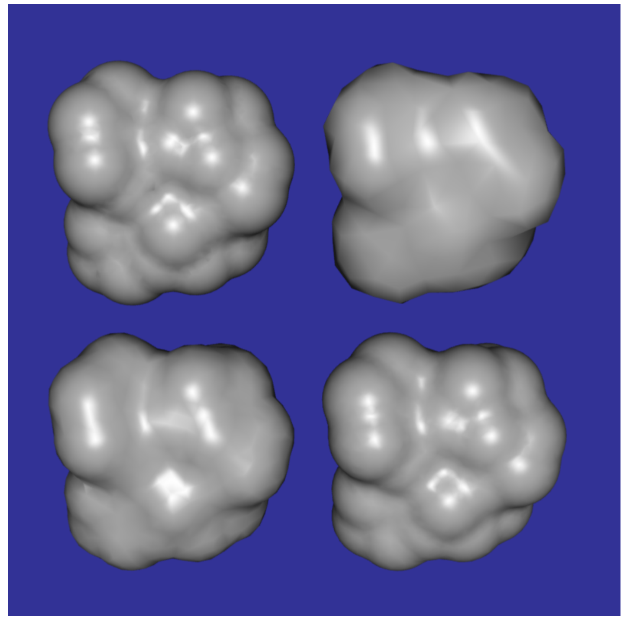


Figure 5: Phong-shaded rendering of the models in Figure 4.

The lower right portion of Figure 1 shows the result of removing all the old vertices from the mutual tessellation that is shown in the lower left. Figure 4 illustrates how several models of various levels of detail can be generated using the above re-tiling method. The model in the upper left of Figure 4 is the original model, a polygonal representation called a Connolly surface of a manufactured carbohydrate. The models shown in the upper right, lower left and lower right are increasingly more detailed re-tilings of the original surface, and they contain 201, 801 and 3676 vertices respectively. The original model contains 3675 vertices. Notice that the most detailed re-tiled version of the model has more evenly-sized polygons than the original model. This demonstrates how successful the point-repulsion method is at placing points uniformly over a surface. Figure 5 shows a Phong-shaded rendering of the same four models.

4.4 Topological Consistency Checks

As mentioned above, two checks must be made before removing an old vertex. If either of these two tests fail, then the vertex R must not be removed. A failing of either check is not a failing of the algorithm, but is instead an indication that the vertex R needs to be retained in the re-tiled model to faithfully represent the topology of the original surface. In practice, nearly all old vertices can be safely removed from a mutual tessellation. The first check is to see that the edges in the set E do not intersect one another except at their endpoints (the vertices in V) when projected into the plane for triangulation. If any pair of these edges do intersect, then R is not removed. This check assures us that the planar triangulation of the points in V will not fold the surface near R. If this check fails we can try projecting the neighborhood of R onto planes at other orientations to see if the edges in E intersect in these cases. If there is a projection onto a plane in which these edges do not intersect, we can remove R and perform the triangulation in this plane. The re-tiling code used to make the images in this paper tries 13 alternate projections before giving up and deciding that a vertex R should be retained.

The second check makes sure that we do not accidentally join the portion of the surface surrounding R to another portion of the surface in front of or behind this region. This can occur when three or four polygons form a narrow neck-like region. For example, Figure 6 shows an old vertex R that we want to remove and the solid lines show the triangles that surround R. The dotted triangles *BDF* and *BDG* are two triangles that form a portion of the surface on the other side of the model. Imagine that the six edges radiating from R are removed and that the region is triangulated. It is likely that one of the new triangles will have *BD* as an edge, which would cause this edge to be shared by three triangles. It is also possible that the triangulation would create the triangle *BDF*, so that this triangle would be present twice in the model. Neither of these situations should be permitted because they would change the topology of the surface. The potential for these problems can be checked by examining triangles near R before the triangulation. If a situation like that of Figure 6 is detected, then the region surrounding R is left alone. This is the approach used to create the re-tiled models in this paper. Another solution is to triangulate the region surrounding R and then see if any of the new triangles would lead to a change in surface topology. If they would, then the vertex R is retained.

4.5 Triangle Shape

There is one additional, optional step that may be performed to assure that the triangles in the re-tiling are well-shaped. This clean-up step examines each vertex of the re-tiled model and attempts to re-triangulate in its neighborhood. This is similar to the vertex removal stage, except that the vertex is not removed but rather is included in the re-triangulation. Figure 7a shows the triangles surrounding a vertex Q whose neighboring vertices are examined during the clean-

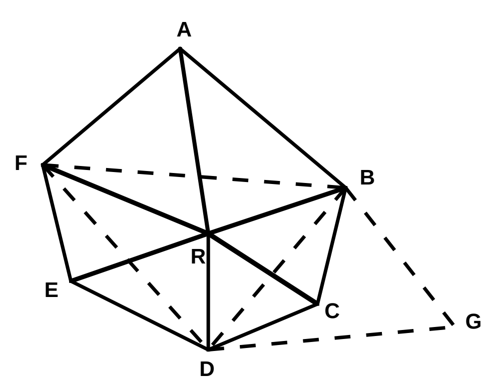


Figure 6: Problem that must be checked during vertex removal.

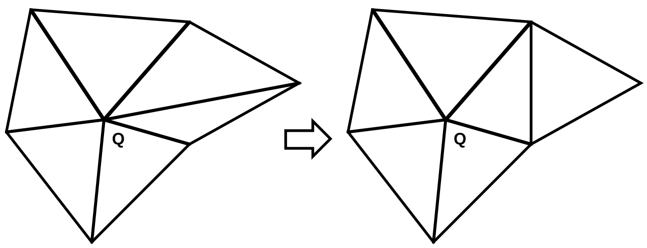


Figure 7: Improving triangle shape by local re-triangulation.

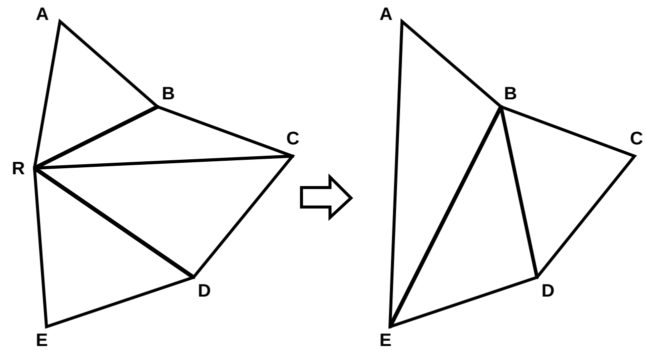


Figure 8: Removing a vertex on the boundary of a surface.

up stage. Figure 7b shows the same vertices, along with *Q*, after local re-triangulation. Any reasonable approximation to a tangent plane to the surface at the vertex *Q* can be used for the plane in which to perform the triangulation. Here again, the same two checks should be performed to avoid re-tiling at a fold and to avoid creating edges shared by three or more triangles. One or two clean-up passes were used in creating all the re-tiled models shown in this paper. The same greedy triangulation routine described earlier was used in this improvement step.

4.6 The Special Case of Boundaries

The re-tiling process can be augmented to handle polygonal models where some of the edges belong to only one polygon. Each of the three steps of mutual tessellation, vertex removal and clean-up must treat these boundary edges specially. When incorporating candidate vertices into a mutual tessellation, any candidate vertex that lies on a boundary edge must be incorporated into the boundary of the polygon that is being triangulated instead of into the interior of the polygon. This means that such a candidate vertex must be an endpoint in two of the constraint edges. During the vertex-removal stage, we choose not to remove an old vertex if it is at the corner of a polygon where two boundary edges meet that are part of the *same* polygon. Likewise, a vertex will be retained if more than two boundary edges meet at that vertex. If exactly two boundary edges from *different* polygons meet at a old vertex, that vertex may be removed. Figure 8a shows such a vertex *R* where the edges *AR* and *RE* belong only to the triangles *ABR* and *RDE*, respectively. Figure 8b shows the triangles formed after removing *R*.

4.7 Re-Tiling Robustness and Extensions

It is worth examining at this point how this re-tiling approach avoids the possible pitfalls involved in connecting candidate vertices. The central strength of the above method is that it breaks the surface re-tiling problem into many small *planar* triangulation problems. Planar triangulation of vertices with constraints is a well-understood problem from computational geometry. Casting the problem into two dimensions avoids the ambiguities found in three dimensions when trying to determine if a point is inside a polygon or whether two edges intersect. Constraining each triangulation sub-problem to include the edges *E* surrounding a vertex *R* that is being removed guarantees us that the collection of newly-created triangles will have the same common boundary as the old triangles *T*. This common boundary is just *E*, the set of constraint edges. These same observations apply to the triangulations performed during the clean-up step. One way to think of the re-tiling process is that the mutual tessellation allows the vertices and polygons of the original model to act as guides for how different portions of the surface will be connected to one another in the re-tiled model.

There is the opportunity within the framework of mutual tessellation and vertex removal to choose a measure of triangle quality for the triangulation sub-problems. The re-tilings shown in this paper were made with a greedy triangulation routine, where the shortest edges that do not intersect already chosen edges are picked to be included in the final triangulation. More specifically, although the triangulation is always performed in a plane, the edge distances used in the greedy algorithm are determined from each vertex's unprojected 3-space position. This greedy algorithm has created well-shaped triangles in the re-tilings that we have performed. If another measure of triangle goodness is desired then another triangulation routine can be incorporated into the basic framework described above. For example, one might use a triangulation routine that attempts to maximize the most acute angle in the potential collection of triangles [De Floriani et al 85].

5 Surface Curvature

5.1 Curvature Approximation

The basic method of point-repulsion gives surface re-tilings in which the new triangles are all roughly the same size across the model. This is quite adequate for surfaces that do not vary greatly in the amount of curvature at different locations. If, however, the variation in surface curvature is relatively large, then the features of the surface would be more accurately reflected in a re-tiling by increasing the density of vertices in regions of high curvature.

Ideally, we would like to have an exact measure of curvature from the object that the polygonal model is meant to represent. Often, however, this information is not available, either because the object being represented is not available (e.g. the volume data was not

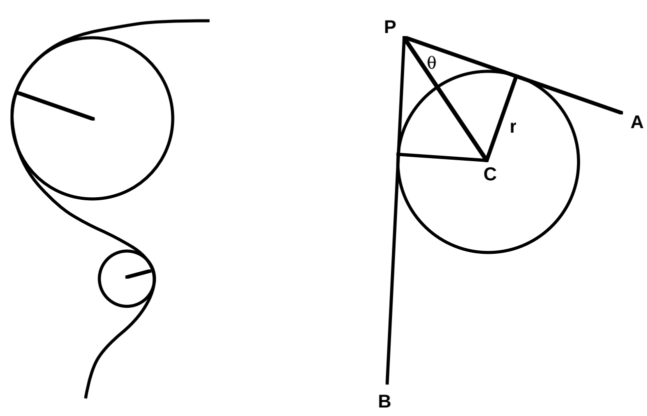


Figure 9: Curvature. (a) Radius of curvature in the plane. (b) Approximation to curvature in the plane at a vertex.

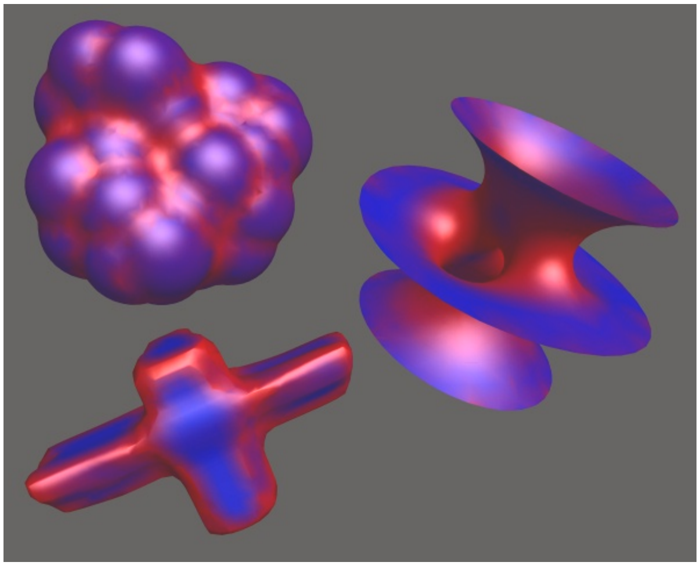


Figure 10: Surface curvature. Red specifies regions of higher curvature and blue shows regions that are relatively flat.

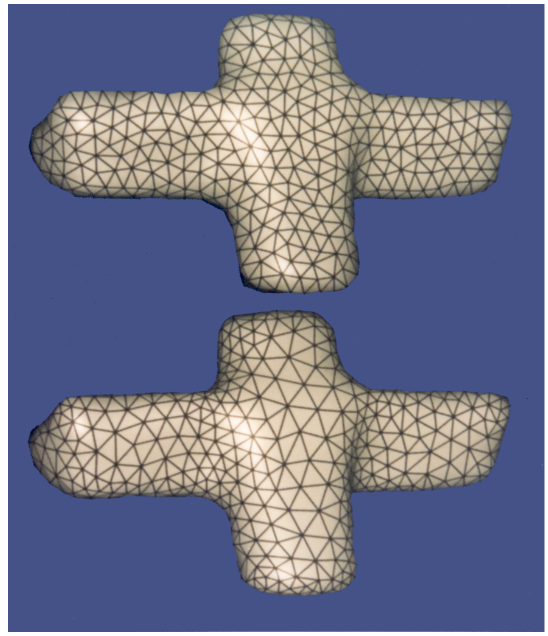


Figure 11: Top surface was created using the same radius of repulsion across the model. Bottom model used curvature to determine the repulsion radius.

retained) or because there never was an exact description of the object (e.g. a cat model was created freehand by a human modeler). For these reasons it is useful to have a way to approximate surface curvature from the polygonal data alone. More precisely, we want to know the maximum principle curvature at any given point on the model. See any text on differential geometry for a mathematical description of principle curvature, such as [O'Neill 66]. Intuitively, this is asking for the radius of the largest sphere that can be placed on the more curved side of the surface at a given point without being held away from the surface by the manner in which the surface curves. Figure 9a shows the radius of curvature at two points along a curve in the plane.

Figure 9b illustrates the curvature approximation used in this paper. This figure shows the two-dimensional version of the curvature estimate near a point *P*. Here a circle has been drawn that is tangent to the edge *PA* at its mid-point and that is also tangent to the longer edge *PB*. The radius of this circle is $r = \tan(\theta)\ |P - A|\ / 2$. In this figure, the line segment *PC* bisects the angle *APB*. This figure will act as a starting point for approximating the curvature of a polygonal surface in 3-space at a vertex *P*.

In the three-dimensional case, the line segment *PC* is replaced by an approximation to the surface normal *N* at the vertex *P*. Then, each edge in the polygon mesh that joins the vertex *P* to another vertex Q_i is examined, and an estimate of the radius of curvature from each of the *n* edges PQ_1, PQ_2,... PQ_n can be computed. Let *V* be the normalized version of the vector $Q_i - P$, that is, a unit vector parallel to the edge PQ_i. Then an estimate for θ_i is $\arccos(N \bullet V)$, and the radius estimate for the edge PQ_i is $r_i = \tan(\theta_i)\ |P - Q_i|\ / 2$. The final estimate *r* of minimum radius of curvature at the vertex *P* is the minimum of all the r_i. This estimate of curvature is a little noisy for some models, so we can smooth the estimate by averaging a vertex's radius *r* with that of all of its neighbors, and we can take this to be the minimum radius of curvature at the vertex. Figure 10 shows the results of this estimate, where each surface is colored red in areas of high curvature (small radius) and is colored blue in the regions that are more nearly flat.

5.2 Concentrating Vertices at Locations of Higher Curvature

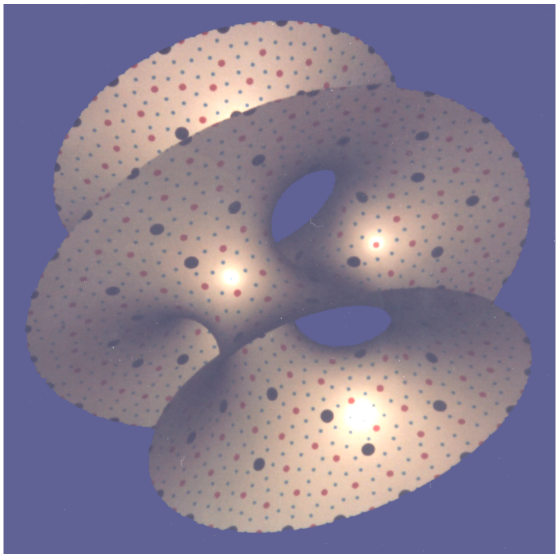
Using the above curvature estimate, we can modify the first step of the point-placement method so that more points are distributed to those places of higher curvature when the points are initially placed on the surface. Recall that the random point placement is area-weighted, so that more points are initially placed on larger polygons. We can increase the density of points on a particular polygon if the polygon's stored value of its area is increased while other polygons' stored area values are held at their correct value. Therefore, to double the density of points on polygons of high curvature, we can multiply the stored area value of these polygons by a factor of two before the area-weighted point-placement step.

Armed with an estimate of curvature over the surface, we can use this value to choose the radius of repulsion in the point-placement routine. We want points that are at very curved areas (small radius of curvature) to push less on their nearby points than points that are on nearly flat regions. This will result in placing more points at the more curved areas. The curvature-adjusted radius of repulsion for a point can be derived from an average of the curvature measures at each of the vertices of the polygon that the point is on. This average is weighted by the distance of the point from each of the polygon's vertices. When computing the force between two points close to one another, the average of their curvature-adjusted radii of repulsion is used instead of using one fixed radius of repulsion for all points. The top portion of Figure 11 shows the re-tiling from 800 points distributed by using the same repulsive radius over all points, and the bottom portion shows the re-tiling given by 800 points that were distributed using curvature-weighted repulsive radii.

6 Interpolation Between Models

6.1 Nested Models

There is a natural nesting of levels of detail in polygonal datasets that are arranged in a rectangular grid of cells, and this nesting of levels can be used to smoothly interpolate between the different levels of detail. If the most detailed version of a terrain model is arranged in



Figrure 12: Spots on minimal surface show the positions of three nested candidate vertex sets.

a 256×256 array of cells, then a 128×128 version of the data can be made by sampling the data at every other vertex of the original cell mesh in each of the x and y directions. Each of the vertices in this reduced grid is also present in the more detailed grid. This section describes how a similar nesting of levels of detail can be made when re-tiling *arbitrary* polygonal models and how we can smoothly interpolate from one level of detail to another. The technique we will use to interpolate between the levels of detail is to flatten some of the vertices and triangles of a higher-detailed model onto the triangles of a model with less detail.

Assume we have a detailed polygonal model and we wish to create three versions of this model that contain 200, 800 and 3200 vertices, and that we want all the vertices in the lower-detailed models to be present in the models with more detail. The first step is to position 200 points on the original polygonal surface using point-repulsion. The 800 vertex model can be created by fixing the positions of the first 200 points, then placing 600 additional points on the object's surface and finally by allowing these new points to be repelled by one another as well as by the 200 fixed points. The most detailed model is then made by fixing these 800 vertices and adding 2400 more in the same manner in which we added the previous 600. Now we have 200 vertices that have the same position in all three models and 800 vertices that are present in the same location in two of the models.

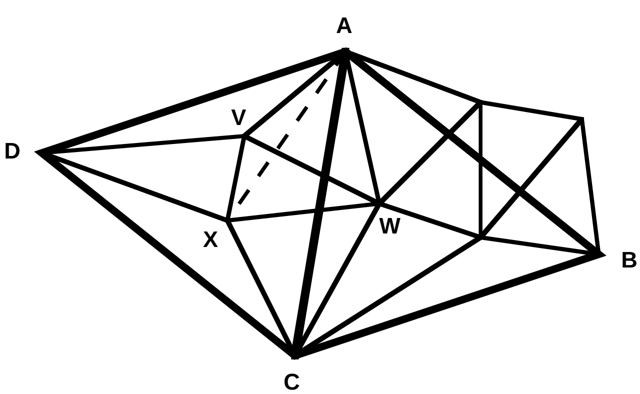


Figure 13: Fragment tracking when removing the high-detail vertices from a model.

Figure 12 shows the positions of the points from three such levels of detail that were created in the manner just described. The large black spots are the 200 initial points, the red spots are the 600 additional points, and the cyan spots are the final 2400 points. The original object is a portion of a minimal surface of mathematical interest that was modelled using 2040 vertices. The spots in this figure were rendered by changing the color at a given surface position if it is inside a sphere centered at one of the 3200 points. Now the issue is to determine how to interpolate between pairs of these models.

6.2 Polygon Fragment Tracking for Interpolation

There are two sub-tasks involved in deciding how to interpolate between a high- and a low-detail model. First, for each vertex V that is present only in the high-detail model, we need to choose a triangle in the low-detail model onto which V may be flattened. Once such a triangle is determined for each such vertex V, we must split each triangle T from the high-detail model by each edge in the low detail model that intersects T. Figure 13 shows a high-detail and a low-detail model drawn together. Vertices A, B, C and D belong to the low-detail model and the edges AB, BC, AC, CD and DA are the edges that will be formed in the low-detail model. These same vertices A, B, C and D are also part of the initial high-detail model. The thinner edges in this figure are the edges of triangles in the high-detail model, and the vertices V, W and X are three of the vertices that are only present in the high-detail model. We require a way of determining that the vertex V can be flattened onto the low detail triangle ACD and that W can be flattened onto ABC. We also want to learn that the high-detail triangle AWV crosses the low-detail edge AC, so that we can split AWV into two triangles for later use in the interpolation procedure. The way to determine this information is to track each vertex such as V and each triangle such as AWV through the entire process of vertex removal as we change the high-detail model into the low-detail model.

The nested point sets shown on the surface in Figure 11 were created by moving from low to high detail. That is, first the low-detail points were placed, then the next higher level, etc. This process is now reversed by working from the high-detail model down to the low-detail model to provide the information we will need to flatten the high-detail triangles on the low-detail model. We begin with the vertices and triangles of the high-detail model and track how some triangles are split and re-formed when the high-detail vertices are removed from the model. Call the set of triangles in the high-detail model H, and let L denote the set of low-detail triangles that make up the model we are working towards. Ultimately, each triangle in L will

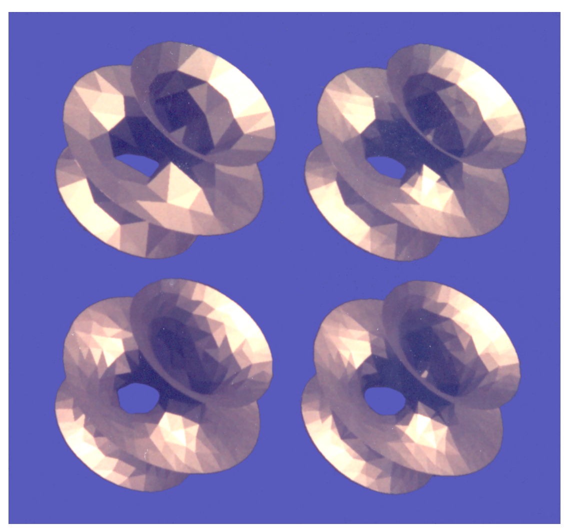


Figure 14: Clockwise from upper left is smooth interpolation between low-detail and high-detail models.

have a pointer to a list of polygon fragments of several triangles from *H*. These polygon fragments retain the positions of their original 3-space vertices from when they were a part of the high-detail model, and they also save their final, flattened position on one of the triangles in *L*. Each of these polygon fragments will also remember what original high-detail triangle they descended from, and this tag will be used to determine which fragments may be re-united after the process of vertex removal.

The polygon tracking process begins as follows. First, each triangle in the high-detail model is initialized with a list of polygon fragments that contains just a single element, and this element is a copy of the original triangle. Each triangle also retains a list of flattened points which is initially empty. Several steps are followed when a high-detail vertex is removed from the model, and these steps all take place in the plane. This lets us unambiguously determine where one edge intersects another and when a point is inside a given triangle. First, when a high-detail vertex *V* is removed then the triangles surrounding *V* must be split by the new edges that are introduced by the re-triangulation of the area surrounding *V*. For instance, assume that the four triangles surrounding *V* in Figure 13 will be replaced by two triangles that share a new edges *AX* (dotted line). This new edge splits each of the old triangles *AWV* and *VWX* into two pieces. The new triangle *AWX* is given a list containing the two polygon fragments that lie within *AWX*. Similarly, the new triangle *AXD* keeps the other two fragments and the undivided polygons *AVD* and *VXD* in its list. Now we must determine which of the new triangles *AWX* or *AXD* the old vertex *V* should be flattened onto. In this example, the new, flattened position of *V* is on triangle *AXD*. This same process of vertex removal, triangle splitting and vertex flattening is carried out for all the high-detail vertices in the model. The result is a set of low-detail triangles *L*, each of which has a list of fragments from the original triangles of the high-detail model. Some of the fragments in a list may be fragments of the same triangle of the high-detail model, and such fragments may be coalesced to give fewer final polygons. Such sibling fragments are found in the same list when a polygon is split at an early stage in the fragment tracking process by an edge that is later removed from the model.

6.3 Performing the Interpolation

When the above work is finished, we have a large collection of polygon fragments that know where they came from in the original, high-detail model and that also know what their current, flattened position is on the surface of the low-detail model. It is now a simple task to interpolate the vertices of each of the polygon fragments between these two positions. At one end of the interpolation they will all lie flat on the low-detail model, and together they will have exactly the same shape as the low-detail model. At the other end of the interpolation, they have a shape identical with the high-detail model. The process of interpolating between these two positions has the effect of "inflating" the low-detail model into the model with more triangles. We have found that linear interpolation between these two positions is sufficient to make smooth transitions between models. There are no jumps or discontinuities during this interpolation. This provides a seamless way of switching from one level of detail to another, and could be useful in both in interactive applications and for rendering frames for animating a complex scene. Figure 14 shows this form of shape interpolation between two models that are re-tiled versions of the minimal surface shown in Figure 12.

7 An Application of Re-Tiling

We have immediate plans to use re-tilings of polygonal surfaces in research on radiation-treatment planning being done at the University of North Carolina at Chapel Hill. Planning the placement of radiation beams for the treatment of tumors is an intensely geometric task [Chung 92]. The problem is how to aim several radiation beams at a tumor while at the same time keeping too much radiation from impinging on the organs surrounding the tumor. James Chung has prototyped a beam-placement application program where the user wears a head-mounted display and places radiation beams around a polygonal representation of the anatomy containing the tumor. The models used to represent the tumor and the surrounding organs (lungs, kidneys, etc.) often contain many thousands of polygons. There is a trade-off that can be made between the accuracy of representation of the anatomy and the frame update rate of the display. One possible solution is to give the user direct control over this update rate [Holloway 91]. The graphics engine would use a more coarse set of polygonal models when the user wishes to make broad motions (e.g. walking around the simulated patient) and would then switch to the more detailed models when fine adjustments are being made to the final beam placements. We plan to make use of the re-tiling techniques described here to provide the variously detailed models.

8 Future Work

One possible extension of the re-tiling method would be to use information about the *direction* of minimum and maximum curvature at each point to help guide the local re-triangulation of the surface. The point-repulsion step could take direction of higher curvature into account by having the points repel in a direction-dependent manner. This would amount to changing the shape of a point's field of repulsion from a circle to an ellipse. The directional curvature measure should also guide which edges between points are created during triangulation. Polygon edges should be created preferentially along the direction of lesser curvature.

There are several more broad issues that should be addressed in future work on re-tiling of polygonal models. One issue is whether there are better ways to estimate the surface curvature on a polygonal model. Another topic is finding measures of how closely matched a given re-tiling is to the original model. Can such a quality measure be used to guide the re-tiling process? Perhaps the biggest issue to explore is the opportunity for elimination of features at very low levels of detail. How can small features of a model be automatically identified and under what conditions is it acceptable to remove a feature completely from a model? For example, no triangles need to be used to represent the shape of a person's ear if the size of the person in the final image will be three pixels high.

Acknowledgments

Many people provided ideas, aid and encouragement for this work, and these people include: David Banks, Henry Fuchs, Marc Olano, Penny Rheingans, John Rhoades, Brice Tebbs and Terry Yoo. Several of the anonymous reviewers made excellent suggestions for improving this paper. Thanks also goes to Kevin Novins and Michael Zyda for discussions about other work that has been done in this area. The radiation dose volume data was provided by the UNC Department of Radiation Oncology and the iso-dose surface was created by Victoria Interrante and James Chung. The molecular model of the carbohydrate called "Wilma" was provided by Mark Zottola. The model of the Costa genus one minimal surface was created by James T. Hoffman using his adaptive mesh algorithm and the mathematical description of this surface is due to Celso Costa, David Hoffman and William Meeks III.

This work was supported by a graduate fellowship from IBM and by the Pixel-Planes Project. Pixel-Planes is funded by DARPA Grant No. DAEA 18-90-C-0044, NSF Cooperative Agreement No. ASC 8920219 and ONR Grant No. N00014-86-K-0680.

References

[Chung 92] Chung, James C., "A Comparison of Head-Tracked and Non-Head-Tracked Steering Modes in the Targeting of Radiotherapy Treatment Beams," *1992 Symposium on Interactive 3D Graphics*, Cambridge, Massachusetts, 29 March - 1 April 1992, pp. 193-196.

[Clark 76] Clark, James H., "Hierarchical Geometric Models for Visible Surface Algorithms," *Communications of the ACM*, Vol. 19, No. 10, pp. 547-554.

[Cosman & Schumacker 81] Cosman, M. and R. Schumacker, "System Strategies to Optimize CIG Image Content," *Proceedings of the Image II Conference*, Scottsdale, Arizona, 10-12 June, 1981.

[Crow 82] Crow, Franklin C., "A More Flexible Image Generation Environment," *Computer Graphics*, Vol. 16, No. 3 (SIGGRAPH '82), pp. 9-18.

[De Floriani et al 85] De Floriani, L., B. Falcidieno and C. Pienovi, "Delaunay-based Representations of Surfaces Defined over Arbitrarily Shaped Domains," *Computer Vision, Graphics and Image Processing*, Vol. 32, pp. 127-140.

[DeHaemer & Zyda 91] DeHaemer, Michael J., Jr. and Michael J. Zyda, "Simplification of Objects Rendered by Polygonal Approximations," *Computers & Graphics*, Vol. 15, No. 2, pp. 175-184.

[Ho-Le 88] Ho-Le, K.," Finite Element Mesh Generation Methods: A Review and Classification," *Computer Aided Design*, Vol. 20, No. 1, pp. 27-38.

[Holloway 91] Holloway, Richard, untitled technical presentation, University of North Carolina at Chapel Hill, December 1991.

[Novins 92] Novins, Kevin, personal communication.

[O'Neill 66] O'Neill, Barrett, *Elementary Differential Geometry*, Academic Press, 1966, New York.

[Preparata & Shamos 85] Preparata, Franco P. and Michael Ian Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, 1985, New York.

[Schmitt et al 86] Schmitt, Francis J. M., Brian A. Barsky and Wen-Hui Du, "An Adaptive Subdivision Method for Surface-Fitting from Sampled Data," *Computer Graphics*, Vol. 20, No. 4 (SIGGRAPH 86), pp. 179-188.

[Schroeder et al 92] Schroeder, William J., Jonathan A. Zarge and William E. Lorensen, "Decimation of Triangle Meshes," *Computer Graphics*, Vol. 26 (SIGGRAPH 92, these proceedings).

[Turk 91] Turk, Greg, "Generating Textures on Arbitrary Surfaces Using Reaction-Diffusion," *Computer Graphics*, Vol. 25, No. 4 (SIGGRAPH 91) pp. 289-298.

[Williams 83] Williams, Lance, "Pyramidal Parametrics," *Computer Graphics*, Vol. 17, No. 3 (SIGGRAPH 83), pp. 1-10.

[Zimmerman 87] Zimmerman, Stephen A., "Applying Frequency Domain Constructs to a Broad Spectrum of Visual Simulation Problems," Evans & Sutherland Technical Document, Presented at the IMAGE IV Conference, Phoenix, Arizona, 23-26 June, 1987.