Mesh Generation from Boundary Models with Parametric Face Representation

Reinhard Klein, Wolfgang Straßer
Universität Tübingen
Bundesrepublik Deutschland [*]

Abstract

The triangulation of boundary representation geometries (BRep geometries) is neccessary for display generation, stereo-lithography applications and also finite element mesh (FE mesh) generation.

The accuracy of the tesselation is of great significance not only for stereo-lithography and rendering algorithms but also for FE mesh generation, since even minor simplifications of the geometry of the solid can lead to large errors in the FE computation. We therefore turn our attention to a new incremental algorithm for the accurate triangulation of meshes of trimmed parametric surfaces. Instead of triangulating each face seperately, the faces of the solid are glued and triangulated together according to the neighbourhood information of the BRep. By this procedure, glueing together faces with different triangulations is avoided. Another advantage of the algorithm is that the error between the linear approximation of the boundary and the boundary itself is controlled step by step until it lies within a predefined tolerance ϵ. Because the algorithm is fully incremental, later improvements of this tolerance can be easily added. The techniques we used to produce a robust implementation of this algorithm under finite precision arithmetic are reported.

CR Categories and Subject Descriptors: I.3.5 **[Computer Graphics]:** *Computational Geometry and Object Modeling*;

Additional Key Words and Phrases: boundary representation, meshing, incremental delaunay triangulation, stereolithography, rendering, finite elements.

1 Introduction

Automatic discretization of three-dimensional objects is a prerequisite for automating engineering analysis applications in a Solid Modeling environment [23],[3]. In general there are two different approaches for the meshing of BReps with curved faces. Both approaches distinguish between FE nodes representing vertices, and nodes representing points on edges, on faces, and in the interior of a solid.

The first approach can be divided into two steps. The first step generates a tetrahedral mesh from the vertices of the BRep. This tetrahedrization will, in general, not be boundary conforming. In this case, in a second step the tetrahedrization can be made to conform to the boundary by the introduction of extra nodes on the boundary of the solid [26].

The second approach divides the mesh generation into two steps. Firstly a boundary conforming triangulation of the faces of the BRep consisting of well-shaped triangles is generated. In the second step the polyhedral approximation is meshed with tetrahedra. In both steps new nodes are added, firstly on the boundary and then in the interior of the solid.

This paper describes an algorithm for the triangulation of curved boundary surfaces of solid models. In combination with mesh generation algorithms for polyhedra [22], [7], [6], [2] it may also be used for threedimensional meshing of solid models. The edges and triangles of the tetrahedrization's boundary approximate the solid's boundary up to a given error tolerance. Depending on the particular modeler, the faces of a BRep are subsets of quadrics or rational polynomial parametric surfaces, mostly NURBS. A face is delimited from the rest of its supporting surface by its edges, which are defined as one-dimensional curve subsets in the parameter domain of the surface. Such faces are called trimmed surfaces. The algorithm described here can be applied to arbitrarily shaped curved solids defined in a solid modeling system that can produce a BRep consisting of parametric trimmed surfaces including explicit definitions of neighbourhoods. The boundary descriptions are not restricted to rational polynomial surfaces, but also other parametric descriptions.

The aim of the triangulation algorithm for trimmed surfaces is to obtain a triangulation that

- approximates the initial surface within a predefined tolerance ϵ,

[*] Wilhelm–Schickard–Institut für Informatik, Graphisch–Interaktive Systeme, Auf der Morgenstelle 10/C9, 7400 Tübingen, E-mail: reinhard@gris.informatik.uni-tuebingen.de.

- avoids cracks between the triangles of different surface patches and
- delivers a simple method to get multiple resolutions of the approximation,
- and in the case of FE mesh generation the triangulation should consists of well-shaped triangles, that is the triangles should not contain obtuse angles.

A good approximation of the boundary of the solid is an essential part, not even for stereolithography applications, where the quality and accuracy of the plastic models depends directly on this features of the triangulation, but also for FE-generation, because even minor simplifications of the geometry of the model can lead to large errors [29].

After shortly presenting some of the most relevant previous work in the next section, the third section proposes the new incremental algorithm with the discussion of some implementation aspects and finally some results produced by the algorithm are shown.

2 Previous work

Several papers deal with implicit surfaces, e.g. [1],[12], [14]. Their main drawback is that these algorithms are not directly applicable to the parametric case, because of their different mathematical treatment. In the paper by Filip et al. [9] some theorems giving bounds on the maximum deviation between parametric surfaces and linear approximations are postulated. These theorems are used to calculate a piecewise linear approximation of non-trimmed patches by uniformly discretizing in u and v directions in the parametric plane. The idea to triangulate the parametric domain of the surface and lift this triangulation to the surface itself is used in most of the algorithms that deal with the problem of approximating parametric surfaces, including the new incremental algorithm presented here. While the algorithm of Filip et. al does not take care of the curvature distribution in the patch, succeeding algorithms presented in the papers of Löhner and Parikh [20], Sheng and Hirsch [27], Shimada and Gossard [28], Chew [4] and Vigo and Brunet [30] take into account the shape of the surface within the algorithm.

It should be noted that except the algorithm proposed by Chew, all these algorithms are not completely adaptive and none of them is incremental. Thus, later refinement of the triangulation is difficult. A drawback of the algorithm proposed by Chew is that there is no direct control of the approximation error. Furthermore there is need of a starting triangulation in that the surface normals do vary by more than $\frac{\Pi}{2}$ in a region about each triangle. In all algorithms the approximation error is controlled implicitly by the maximum length of the triangle sides. In the algorithm of Vigo and Brunet the length of the triangle sides can vary over the trimmed surface patch and so an adaptive triangulation can be generated. In the algorithm of Sheng and Hirsch global bounds for each patch of the boundary of the solid are used. This has the disadvantage of being not adaptive. The number of required triangulation vertices is too large. Shimada and Gossard generate adaptive 2D/3D meshes for a given boundary representation of the geometry of a space region and a density distribution over that region. The method uses a physically-based approach that simulates the attraction and repulsion forces between bubbles covering the region. Although it guarantees a conformal triangulation with well-shaped triangles, solving the equation system can become inefficient.

3 Basic definitions, the triangulation algorithm

The BRep is based on the observation that a solid can be completely defined by its boundary elements: the faces, edges and vertices. A vertex is a point in 3-space represented by its coordinates. An edge is a part of a line or curve bounded by two vertices. It is represented by the equation for the line or curve and references to its two bounding vertices. A face is a bounded part of a surface. It is represented by the equations for the surface and bounding loops, where a loop is a closed sequence of edges.

A BRep uses a graph as data structure. The boundary elements are represented in nodes. The links between the nodes represent the adjacency relationships between the boundary elements [21], [13].

If we represent the faces by trimmed parametric surface patches the edges of the bounding loops are defined as curves in the parameter domain of the supporting surfaces. The following definition of a planar domain follows the one given in [26].

Definition 3.1 (Planar domain) *A* planar domain $\Omega \subset \mathbb{R}^2$ *is defined by a set* $B = \{b_0, \cdots, b_N\}$ *of* boundary polygons. *The set* B *includes an exterior boundary* b_0 *and interior boundaries (holes)* $b_1, \cdots b_N$. *Each boundary* b_i *consists of a finite number* (≥ 3) *of oriented curved* domain edges $e_{i0}, \cdots, e_{in_i}$ *and is defined by a set of* boundary nodes $V \supset V_i = \{v_{i0}, \cdots, v_{in_i}\}$. *Every boundary polygon* b_i *defines a region* Ω_i. *The interior of these regions* $\overset{\circ}{\Omega}_i$ *is located on the left side of the oriented boundary polygons . In order for* Ω *to be a well-defined finite domain, the following conditions must hold true:*

$$\begin{aligned}
\overset{\circ}{\Omega}_i &\subset \overset{\circ}{\Omega}_0,\ 0 \leq i \leq N, \\
\overset{\circ}{\Omega}_i &\cap \overset{\circ}{\Omega}_j = \emptyset,\ 1 \leq i, j \leq N,\ i \neq j, \\
b_i &\cap b_j \subset V,\ 0 \leq i, j \leq N,\ i \neq j.
\end{aligned}$$

For simplicity we assume that all edges e_{ij} *are parameterized over the interval* $[0, 1]$.

In Figure 1 an example of a simple planar domain is shown.

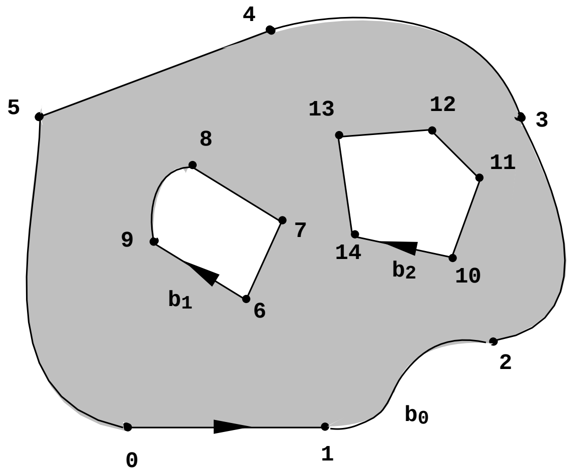

Figure 1: A simple planar domain Ω, described by the outer boundary b_0 and two inner boundaries b_1 and b_2. The edges of the boundaries are oriented in such a way, that the interior of the domain is located on the left side of the edges.

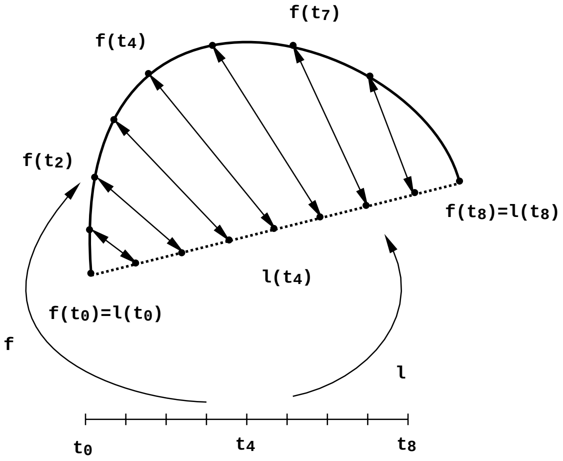

Figure 2: The approximation error of the linear approximation is measured by the sup-norm. For each point in parameter-space the euclidean distance between the corresponding points on the linear approximation and the curve itself is measured.

Definition 3.2 *Let $L(\Omega)$ be an approximation of the planar domain Ω, i.e. let the edges of the boundary be approximated with straight line segments, and $V \subset L(\Omega)$ a set of vertices which contains all vertices $\bigcup_{i=0}^{M} V_i$ of the approximated boundary. A set $T = \{T_i\}_1^t$ of non degenerated, open triangles is a triangulation of $L(\Omega)$ if:*

- *V is the set of all vertices in T.*
- *Every edge of a triangle in T contains only two points from V, namely its endpoints,*
- *$\overline{L(\Omega)} = \bigcup_{i=1}^{t} \overline{T}_i$,*
- *$T_i \cap T_j = \emptyset,\ i \neq j$.*

Let $f : \Omega \longrightarrow \mathbb{R}^3$ be a trimmed parameterized surface, where $\Omega \subset \mathbb{R}^2$ is compact. We are interested in a simultaneous interpolation $L(\Omega)$ of the planar domain Ω and $f_{\mathcal{T}}$ of f from the space $S_1^0(\mathcal{T})$ of piecewise linear functions defined on a triangulation $\mathcal{T} = \{T_i\}_1^t$ of $L(\Omega)$. To get an interpolation $L(\Omega)$ we search for an interpolation $e_{ij_{\mathcal{K}}}$ of the boundary edges e_{ij} from the space $S_1^0(\mathcal{K}_{ij})$ of piecewise linear functions defined on knot sequences $\{k_{ij_l}\} \subset [0,1]$. More precisely the spaces $S_1^0(\mathcal{K})$ and $S_1^0(\mathcal{T})$ are defined by

$$\begin{aligned} S_1^0(\mathcal{K}) &= \{h \in C^0([0,1]) \mid h_{|[k_{ij_l}, k_{ij_{l+1}}]} \in \Pi_1^2\}, \\ S_1^0(\mathcal{T}) &= \{g \in C^0(\Omega) \mid,\ g_{|T_i} \in \Pi_1^3\}, \end{aligned}$$

where Π_1^2 and Π_1^3 is the two (respectively three) dimensional space of linear polynomials.

Definition 3.3 (Approximation error) *The approximation errors $E(f \circ e_{ij_{\mathcal{K}}}, f \circ e_{ij})$ between a trimming curve $f \circ e_{ij}$ and its linear approximation $f \circ e_{ij_{\mathcal{K}}}$ and between a trimmed surface f and a function $f_{\mathcal{T}} \in S_1^0(\mathcal{T})$ are measured by the sup-norm*

$$\begin{aligned} E(f \circ e_{ij_{\mathcal{K}}}, f \circ e_{ij}) &= \|f \circ e_{ij_{\mathcal{K}}} - f \circ e_{ij}\|_\infty \\ &= \sup_{t \in [0,1]} \|f \circ e_{ij_{\mathcal{K}}}(t) - f \circ e_{ij}(t)\|, \end{aligned}$$

$$\begin{aligned} E(f, f_{\mathcal{T}}) &= \|f - f_{\mathcal{T}}\|_\infty \\ &= sup_{(u,v) \in L(\Omega)} \|f(u,v) - f_{\mathcal{T}}(u,v)\|. \end{aligned}$$

The approximation error $E(f, f_{\mathcal{T}})$ measures the maximum parametric distance between the piecewise polygonal approximation $f_{\mathcal{T}}$ and the surface f. For the case of planar curves an example is shown in Figure 3.

Each face of the BRep has its own normalized local (u,v) parametric coordinate system, and all vertices and edges of a face are defined with respect to it. Thus, each vertex and each edge has a local representation for each face to which it belongs. Figure 3 shows two intersecting faces and the relations between their intersection edges in both parameter spaces. The same edge is differently parameterized and there are different approximation errors between a trimming curve and its linear approximation in $\mathbb{R}^3$ with respect to the sup norm, depending on the face to which the curve belongs. Therefore, for each edge of the solid we use the maximum of the different approximation errors of an edge caused by its uses in different neighbouring faces.

Using the above error measure the following remarks on the approximation of the trimming curve and of the surface itself should be taken into account:

1. The approximation error $E(f, f_{\mathcal{T}})$ depends on the approximation of the domain Ω.
2. Let $e_{ij_{\mathcal{K}}}$ be a linear interpolation of the curved boundary edge e_{ij}. Then the approximation error $E(f_{|e_{ij}} - f_{\mathcal{T}})$ and $E(f_{|e_{ij_{\mathcal{K}}}} - f_{\mathcal{T}})$ are in general different, see Figure 4.

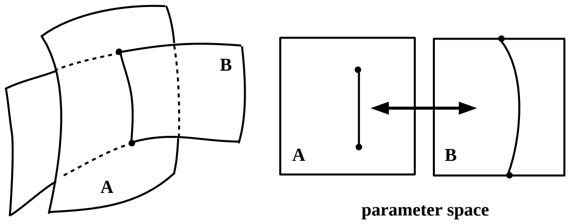

Figure 3: Two intersecting faces and the representation of the intersection curve in both parameter spaces.

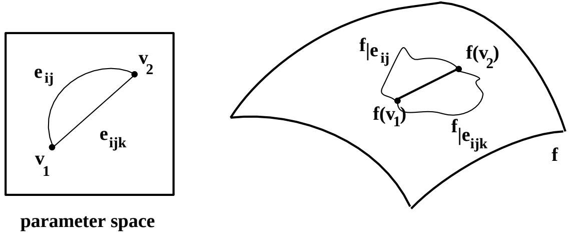

Figure 4: The curves $f_{|e_{ij}}$ and $f_{|e_{ij_{\mathcal{K}}}}$ are different. This results in different errors of these curves with respect to the linear approximation between $f(v_1)$ and $f(v_2)$.

3. The linear approximation of the trimming curves by line segments is performed in Euclidean space instead of in parametric space. Therefore, the tolerance used for the approximation of the trimming curves is exactly the same for the triangulation.

The algorithm The basic idea of the algorithm is to start with an initial Constrained Delaunay triangulation of Ω and to incrementally insert points into the triangulation of the parametric space, one point at each step causing an update of the triangulation. This scheme for adaptive approximation was proposed by several authors (see, e.g., Floriani, Falcidieno, and Pienovi [5], Lee and Schachter [19] and Rippa [25].) in other contexts. The points to insert are chosen under the condition that the corresponding points on the trimming curves or on the surface have maximum distance to the actual linear approximation of the trimming curves or to the actual triangulation. Due to the variety of fast algorithms to insert new points into a Delaunay triangulation and of its optimality properties in our implementation, a Delaunay triangulation is used. In principle any other kind of triangulation can be used instead. The algorithm can be outlined as follows:

- Compute the boundary of the surface including trimming curves in parameter space. Neglect the inner borders between different patches forming a face.

- Generate an initial triangulation of the parameter domain of the faces which contains the vertices on the boundary-curves and on the trimming-curves.

- Insert stepwise a point into the parametric space with the property that the corresponding point on the surface has maximum distance either to a trimming curve on the surface patch or to the triangulation. Stop the insertion if both maximum distances are smaller than a given maximum error bound.

During the algorithm we have to take care of the following case. If a approximation $L(\Omega)$ of the domain Ω is given, it may happen that a point causing the maximum approximation error belongs to $L(\Omega)$ but not to Ω itself. Such a point should not be inserted into the triangulation, see fig 5.

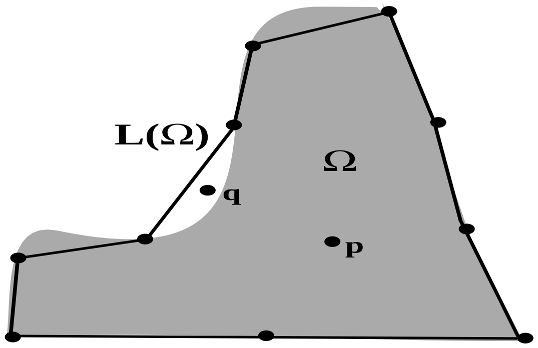

Figure 5: The point q belongs to $L(\Omega)$ but not to Ω.

To avoid these cases we compute the distance to the boundary in parameter space for each point to be inserted. If this distance is greater than the actual approximation error of the nearest boundary edge we refine the boundary edge instead of inserting the point.

In the case that a point on the boundary of a face is inserted, the corresponding points on the boundary of neighbouring faces are also inserted and the triangulations on that faces are updated. Since each edge is represented in parameter space of all faces adjacent to that edge, a procedure was developed to map a point (u_0, v_0) defined in parameter space (u, v) of face F_1, to a point (s_0, t_0) in the parameterspace (s, t) of face F_2. This is implemented by firstly computing the Euclidean point (x_0, y_0, z_0) for the parametric point (u_0, v_0) followed by solving the three algebraic equations in two variables

$$x_0 = x(s,t) \quad y_0 = y(s,t) \quad z_0 = z(s,t).$$

The following theorem ensures that in the case of Lipschitz-continuous functions the algorithm terminates.

Theorem 3.4 *Let $f : \Omega \longrightarrow \mathbb{R}^n$ be a Lipschitz-continuous function on a compact set $\Omega \subset \mathbb{R}^2$, that is, there is a constant $L > 0$, such that*

$$||f(v_1) - f(v_2)|| \leq L||v_1 - v_2|| \;\; \forall\, v_1, v_2 \in \Omega.$$

Then the described algorithm terminates after a finite number of steps.

The proof of the theorem follows immediately from the following lemma since the area of the bounded domain Ω is finite.

Lemma 3.5 *In the situation of the theorem let $V \subset \Omega$ be a finite set of points $v_i,\ i = 0, \cdots N$ and $\mathcal{T} = \{T_i\}_1^n$ a triangulation of V. Let $f_{\mathcal{T}} \in S_0^1(\mathcal{T})$ be a piecewise linear interpolation function of f on $\mathcal{T}$. Suppose that $||f(v) - f_{\mathcal{T}}(v)|| > \delta > 0$ for some $v \in \Omega$. Then we have*

$$||v - v_i|| > \frac{\delta}{2L},$$

$\forall\ v_i \in V,\ i = 0, \cdots, N.$

Proof The difference $f - f_{\mathcal{T}}$ is Lipschitz continuous with Lipschitz constant $2L$. Now we have $\forall v_i \in V, i = 0, \cdots, N$

$$\begin{aligned} \delta &< ||(f - f_{\mathcal{T}})(v) - (f - f_{\mathcal{T}})(v_i)|| \\ &\leq 2L||v - v_i|| \end{aligned}$$

and therefore

$$||v - v_i|| \geq \frac{\delta}{2L}.$$

Thus, after the insertion of a finite set of points the error is less then a given $\delta > 0$.

3.1 Computation of a point with maximum approximation error

The main part of the algorithm is to compute in each step a point that causes the maximum approximation error E of the actual triangulation T and actual approximated domain $L(\Omega)$. We start with a linear approximation $L^0(\Omega)$ and a Delaunay triangulation $\mathcal{T}^0 = \{T_i^0,\ i = 1 \cdots t^0\}$ of the boundary of $L^0(\Omega)$. For each edge $e_{ij_{\mathcal{K}^0}}$ of the boundary of $L^0(\Omega)$ and for each triangle $T_i^0 \in T^0,\ i = 1, \cdots, t^0$ we compute the maximum approximation error between $f \circ e_{ij}$ and $f \circ e_{ij_{\mathcal{K}^0}}$ and $f_{|T_i^0}$ and $f_{\mathcal{T}|T_i^0}$ respectively. The edges $e_{ij_{\mathcal{K}^0}}$ and the triangles T_i^0 together with the point in the parameter domain which causes the maximum error over the edge and over the triangle are held in lists S_Ω and $S_{\mathcal{T}}$ sorted in ascending order by the approximation errors $E(f \circ e_{ij}, f \circ e_{ij_{\mathcal{K}^0}})$ and $E(f_{|T_i^0}, f_{T|T_i^0})$. Triangles and edges with a maximum approximation error $E(f \circ e_{ij}, f \circ e_{ij_{\mathcal{K}^0}})$ and $E(f_{|T_i^0}, f_{T|T_i^0})$ respectively, less than the predefined maximum error bound are not inserted into the lists.

In the first step the topmost point of the list is inserted into the constrained Delaunay-triangulation of Ω. With the knowledge of the triangle containing the point to be inserted this can be done very fast. In average the insertion of a point in the existing constrained Delaunay-triangulation is a local procedure. Only a few triangles in a neighbourhood of the triangle containing the point to be inserted are retriangulated, see Figure 6. After the insertion of the point only the approximation errors between f and the piecewise linear approximation over the new triangles have to be computed and if neccessary inserted into the sorted list S. The approximation errors between f and the piecewise linear approximation over the other triangles remain unchanged in this step. The triangles of the retriangulated area, that do not any more exist in the actual triangulation are deleted from S. Proceeding this way the point $(u, v) \in \Omega$ causing the maximum approximation error is on top of the list after each step.

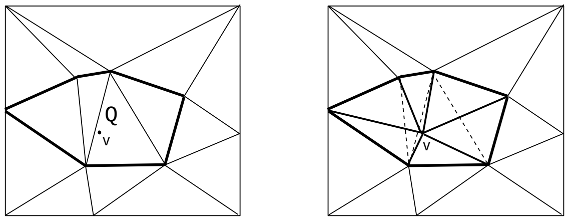

Figure 6: Only a few triangles in a neighbourhood of the triangle containing the vertex v to be inserted are retriangulated. The left side shows the so called influenced area Q of the vertex v. On the right side the retriangulated area with the new triangles and the triangles removed are shown.

For the sequencing triangulations $T^k = \{T_i^k,\ i = 1, \cdots, t^k\}$ the above step is repeated until the sorted list is empty.

If points on the boundary of Ω are inserted the corresponding boundary edges have to be removed and two new edges have to be inserted. In addition the area influenced by the removed edge and the new point have to be retriangulated. For this purpose we developed a special algorithm to remove edges from a Delaunay Triangulation.

Presampling Because of the similarity of the presampling in the case of curves and surfaces it will described only once for the case of surfaces. To compute the maximum triangulation error over the triangles of the actual triangulation the surface is pre-sampled in such a way that a triangulation of the pre-sampled points $(u_j, v_j) \in \Omega\ j = 0, \cdots, p$ in the parameter-domain delivers a piecewise linear approximation f_P of the surface with a maximum approximation error $E(f, f_P) = sup_{(u,v)\in\Omega}||f(u, v) - f_P(u, v)|| < \frac{\epsilon}{2}$ that is smaller than half of the predefined maximum error bound ϵ. Using this piecewise linear approximation of the surface instead of the surface itself and a half of the predefined error bound, the problem of finding a point in the parameter domain which causes a maximum approximation error can be reduced to a comparison of approximation errors of a finite number of sample points and a sequencing linear interpolation between some of those points. For the actual approximation error we have

$$E(f, f_T) \leq E(f, f_P) + E(f_P, f_T) \leq \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon.$$

In this way the problem of finding a point in parameter domain which causes a maximum approximation error becomes a discrete one.

At this point it should be mentioned that presampling is much cheaper than other approaches, such as quasi-Newton methods or conjugate gradient methods, in order to find the point which causes the maximum approximation error. Furthermore the problems of local and global maximums inherent to that kind of methods are avoided.

Regular grids If bounds on the second derivative of the function f are known a regular pre-sampling is done. As already mentioned in [9] we also found that in many cases this is faster then adaptive algorithms. On the resulting regular grids we use an active- edge-list scan-line algorithm which is also used in computer graphics for the rasterization of triangles [10]. In such a way the distance $||f_P(u_j, v_j) - f_T(u_j, v_j)||$ for all sample points $(u_j, v_j) \in T_i$ is computed, see Figure 7.

Quadtrees In other cases, e.g. Bézier-tensorproductsurfaces, B-spline-tensorsurfaces or Bézier-trianglesurfaces, an adaptive pre-sampling based on an estimation of the absolute value of the distance between corresponding points of the linear approximation and the control-points is performed. This leeds to a quadtree-data structure on the parameter domain. Because we are only interested in the approximation error and not in the topology of the quadtree, it is not neccessary that the quadtree fulfills any conditions on neighbouring cells. In our case every neighbour of a cell can be subdivided several times, more or less, as the cell itself.

We use a linear coding of the quadtree [11], see Figure 8. For each point of the actual approximative triangulation the address of the quadtree-cell containing the point is stored. In this way the smallest cell containing a triangle of the actual triangulation and its sample points can be found very fast. An additional flag prevents, that the error at the corner-points of the cells is computed twice.

Composite surfaces In many applications faces are built from several surface patches, such as nets of B-Spline-tensorproductsurfacepatches or Bézier-triangle-patches. Given a domain $\Omega = \cup_{i=1\cdots N}\Omega_i$ as the disjunct union of domains Ω_i and a set of parameterized functions $f_i : \Omega_i \mapsto \mathbb{R}^3$, $i = 1, \cdots, N$ a composite parameterized surface $F : \Omega \mapsto \mathbb{R}^3$ is defined by $F(p) = F_i(p)$ if $p \in \Omega_i$. The functions f_i are allowed to be of different types. For example, a composite surface can contain a B-spline-tensorproduct surface, a Bézier-triangle-surface and a B-spline-triangle-surface at the same time. For each type of patch different sampling strategies, regular or quadtree, may be appropriate. Therefore the described algorithm to find the point which causes the maximum approximation error has to take care of this situation. For triangles belonging to different subdomains Ω_{i_k} the corresponding parts are processed individually with the algorithm needed for the sampling strategy on that patch. To find for every sample point $(u, v) \in \Omega$ the corresponding domain Ω_i, the domain Ω is triangulated and the triangles are administrated in a quadtree-structure. Each cell contains a list of triangles incident to the cell. This quadtree-structure is independent from the quadtree-structures generated by the sampling of the different patches. If a point $(u, v) \in \Omega$ is given the corresponding quadtree-cell is determined. For the triangles incident to the cell a point in triangle test is performed. The implementation is simplified using object oriented methods. Details are described in [15], [17].

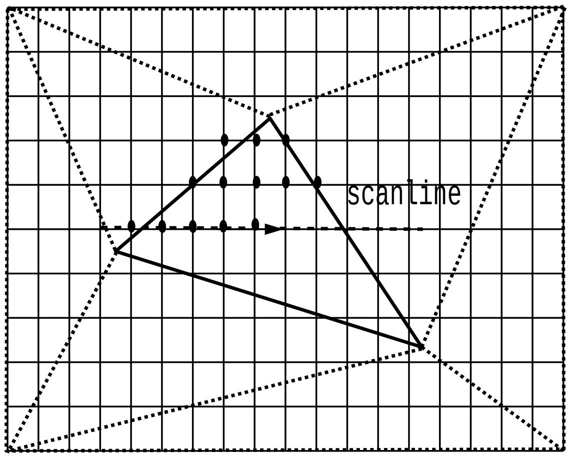

Figure 7: On a regular grid a scan-line algorithm is used.

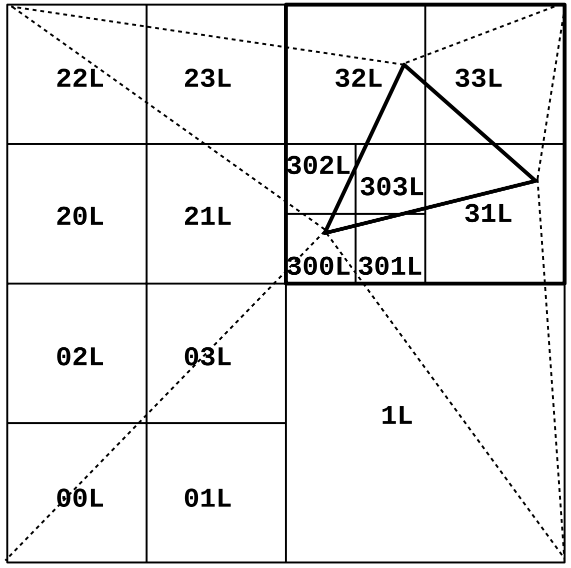

Figure 8: Using a linear coding for the quadtree it is easy to find the sample points corresponding to a given triangle. The address of the bounding quadtree-cell of the triangle is $3L$.

3.2 Multiple points with same maximum approximation error

Points that cause a maximum approximation error may not be unique, for example in the case of cylinders. In this situation we use a simple but efficient heuristic. All points which cause the same approximation error are gathered. For each of these points the number of border elements of the influenced area, that is the number of new triangles that will be created by inserting the point are computed, see Figure 6. Then the point is chosen, for which a minimum number of new triangles will be generated by the insertion procedure. In the case of surfaces like cylinders, cones, etc, for which in each step a point on a boundary curve is among the points with maximum approximation error, the point on the boundary will be inserted.

3.3 Different levels of approximation errors

For the purpose of visualization we are interested in a dynamic change of the quality of the linear approximation to the surface. If an object is far away from the viewer, a coarse approximation is sufficient. If the object comes nearer to the viewpoint, a better approximation is needed. The triangulation produced by the algorithm is well suited for this case. Because the vertices of the triangulation are inserted step by step, there is a natural ordering of the vertices $v \in V$ by insertion time. After the insertion of the vertex v in the l-th step, the maximum approximation error $E(f, f_{T_l})$ between the triangulation T^l and the surface f is known. In addition to the vertex v this error is stored in a common structure. In a preprocessing step a piecewise linear approximation with a predefined minimal approximation error is computed and the sorted list of vertices together with their approximation errors is stored. Afterwards, the topology of the corresponding triangulation T is implicitly given by the use of the Delaunay-triangulation.

If different levels of approximation are needed, the different approximations can be calculated by inserting or removing step by step the vertices of the sorted list into the constrained Delaunay triangulation of the domain Ω. In each step the actual approximation error is known. The insertion of vertices into a constrained Delaunay-triangulation can be done very fast, see [16]. Using special acceleration structures we are able to insert 1000 vertices in about 35 msec into a Delaunay-triangulation on SGI/Indigo with R4000 processor. After the preprocessing, triangulations of the surface with different approximation levels can be generated in real-time.

4 Remarks on the use of the Delaunay-triangulation

In [8] it is documented that linear approximation over a triangle can be improved choosing the shape of the triangle according to the specific behavior of the underlying function. Though minimizing a certain roughness criteria of the triangulation [24] the Delaunay-triangulation is not always the best choice. For the same sample points of the surface there may exist another triangulation with a lower approximation error. As shown in [8] such triangulations can be computed using a local optimization procedure (LOP) proposed by Lawson [18]. Nevertheless in the algorithm the use of the Delaunay-triangulation has significant advantages:

- Only a small number of triangles is influenced by the insertion of a point into the triangulation. Therefore the amount of calculations needed to find the point causing the maximum error is restricted.
- There are elaborated and fast algorithms to insert points into a Delaunay-triangulation.
- After calculating the sequence of inserted points together with their corresponding triangulation errors, up to a certain approximation tolerance, constraint Delaunay-triangulations with different levels of approximation errors, can be generated in real time.

5 Future work

In general, at least at the beginning of the algorithm, the parameterization of the starting triangulation and the parameterization of the surface itself is different. Using the error-measure described above would balance this difference. After the termination of the algorithm, the parameterization of the linear approximating surface and the parameterization of the surface itself is approximately the same. But one may also be interested in a linear approximation where the approximation error is measured by the Haussdorff-distance. The Haussdorff-distance is defined by

$$h(f, f_T) = \max(h_{ff_T}, h_{f_T f}),$$

where $h_{ff_T} = \sup_{x \in f(\Omega)} \inf_{y \in f_T(\Omega)} d(x, y)$ and $h_{f_T f} = \sup_{y \in f_T(\Omega)} \inf_{x \in f(\Omega)} d(x, y))$. It is always smaller than the sup-norm used in the algorithm presented here. We are working on an approximative triangulation based on this distance.

Further one should find fast algorithms and techniques to insert points into other triangulations, instead of Delaunay-Triangulations which influence the actual triangulation only on a small number of triangles.

Last not least we are working on an extension of this algorithm to get better control of the shapes of the resulting triangles. In this way the algorithm will become more suitable for FE mesh generation.

6 Conclusions

A new incremental algorithm for adaptive piecewise linear approximation of the boundary of BReps by triangles has been presented. In combination with tetrahedrization algorithms it can be used for threedimensional meshing of solid models. The algorithm is able to triangulate faces composed of several patches. Furthermore, the maximum approximation error can freely be chosen by the user. In each step

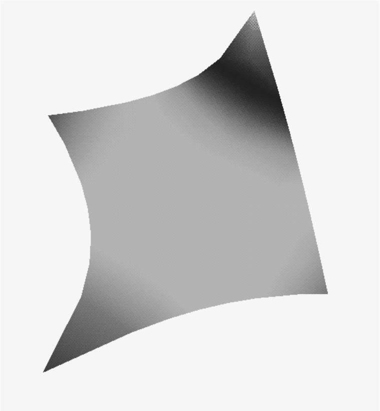

Figure 9: A B-spline-tensorproduct patch.

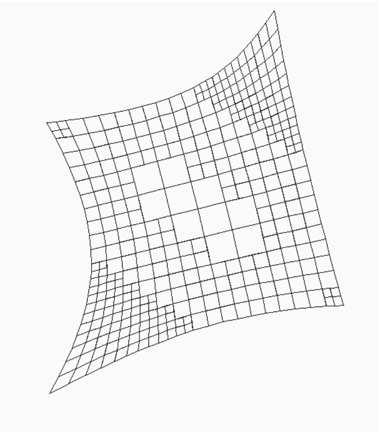

Figure 10: Adaptive pre-sampling of the patch.

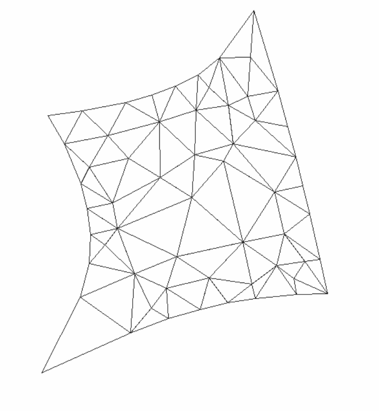

Figure 11: The resulting triangulation.

of the algorithm the maximum approximation error of the actual triangulation and its boundary is known. Due to the incremental nature of the algorithm further refinement of the boundary triangulation can be easily performed in order to get a better approximation. In the Figures 9-11 a tensor-product Bézier-patch, the adaptive pre-sampling of the patch and the final triangulation are shown. Figures 12 shows a adaptive triangulation of the boundary of a teapot consisting of several trimmed faces.

7 Acknowledgement

We would like to thank T. Hüttner for his efforts on all the major implementation issues and the fruitful discussions that led to the realization of this algorithm.

References

[1] ALLGOWER, E., AND GNUTZMANN, S. Simplicial pivoting for mesh generation of implicitly defined surfaces. *Computer Aided Geometric Design 8*, 4 (1991), 305–326.

[2] BAKER, T. Automatic mesh generation for complex three-dimensional regions using a constrained delaunay triangulation. *Engineering with Computers 5* (1989), 161–175.

[3] CAVENDISH, J. C., FIELD, D. A., AND FREY, W. H. An approach to automatic three-dimensional finite element mesh generation. *Internat. J. Numer. Methods Eng. 21* (1985), 329–347.

[4] CHEW, L. P. Guaranteed-quality mesh generation for curved surfaces. In *Proceedings of the Ninth Annual ACM Symposium on Computational Geometry* (1993), ACM, ACM Press, pp. 274–280.

[5] DEFLORIANI, L., FALCIDIENO, B., AND PIENOVI, C. Delaunay-based representation of surfaces defined over arbitrarily shaped domains. *Computer Vision, Graphics and Image Processing 32* (1985), 127–140.

[6] DEY, T. Triangulation and CSG representation of polyhedra with arbitrary genus. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.* (1991), pp. 364–372.

[7] DEY, T. K., BAJAJ, C. L., AND SUGIHARA, K. On good triangulations in three dimensions. *Internat. J. Comput. Geom. Appl. 2*, 1 (1992), 75–95.

[8] DYN, N., LEVIN, D., AND RIPPA, S. Data dependent triangulations for piecewise linear interpolation. *IMA J Numer. Analysis 10* (1990), 137–154.

[9] FILIP, D., MAGEDSON, R., AND MARKOT, R. Surface algorithms using bounds on derivatives. *Computer Aided Geometric Design 3*, 4 (1986), 295–311.

[10] FOLEY, J. D., VAN DAM, A., FEINER, S. K., AND HUGHES, J. F. *Fundamentals of Interactive Computer Graphics*, second ed. Addison-Wesley Publishing Company, 1990.

[11] GARGANTINI, I., AND TABAKMAN, Z. Linear quad- and oct-trees: Their use in generating simple algorithms for image processing. In *Proceedings of Graphics Interface '82* (1982), K. B. Evans and E. M. Kidd, Eds., pp. 123–126.

[12] HALL, M., AND WARREN, J. Adaptive polygonalization of implicitly defined surfaces. *IEEE Computer Graphics and Applications 10*, 6 (Nov. 1990), 33–42.

[13] HOFFMANN, C. *Geometric and Solid Modeling An Introduction*. Morgan Kaufmann Publishers, Inc., 1989.

[14] KLEIN, R. Polygonalization of real algebraic surfaces. Accepted for the second international Conference on Curves and Surfaces, Chamonix, June 1993.

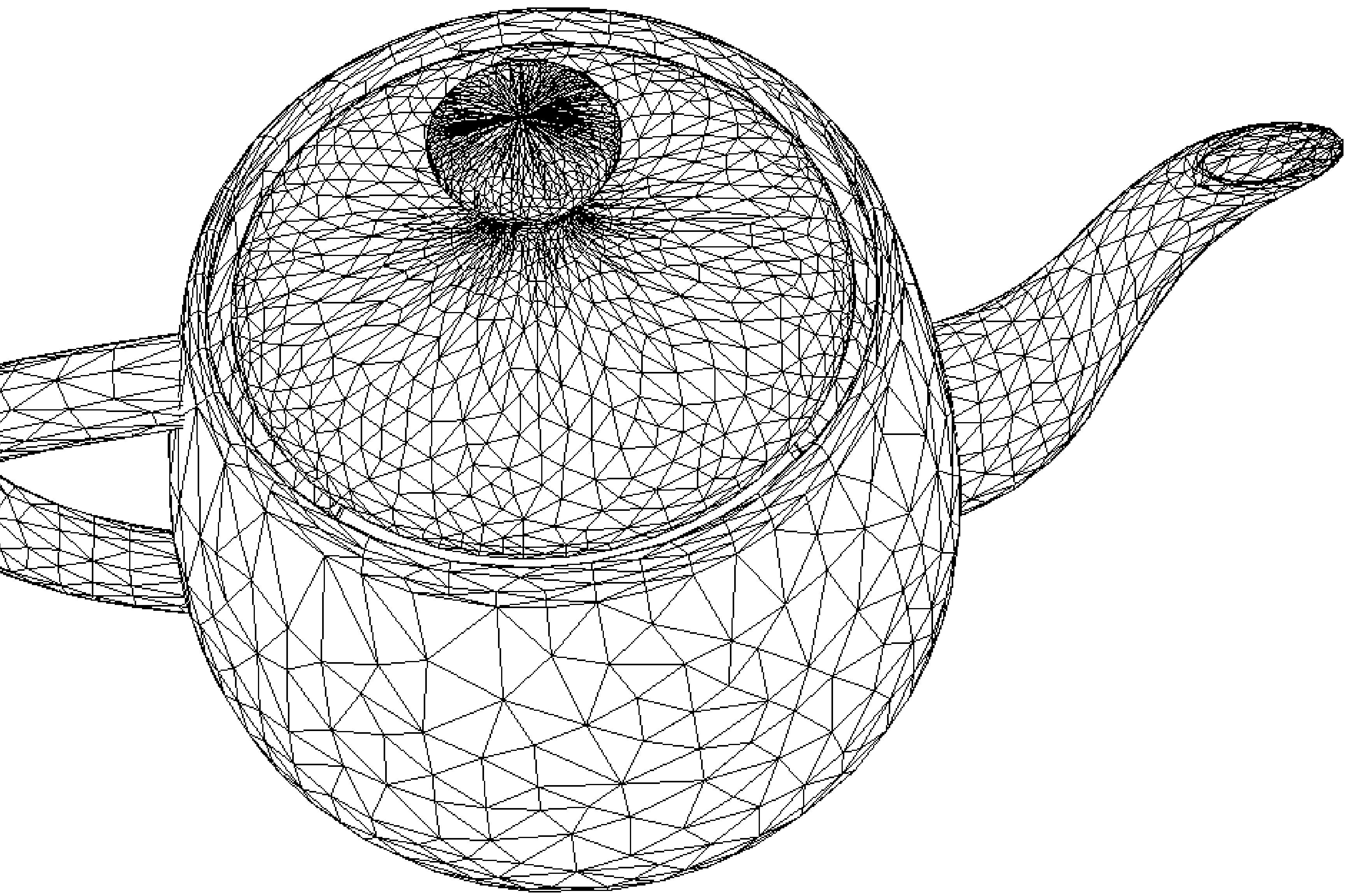

Figure 12: Triangulation of the Utah teapot.

[15] Klein, R. Surfaces in an object-oriented geometric framework. In *Graphics and Robotics*, W. S. er and F. Wahl, Eds. Springer Verlag, 1993.

[16] Klein, R., and Krämer, J. Delaunay triangulations of planar domains. Tech. rep., Wilhelm-Schickard-Institut, Graphisch Interaktive Systeme, Universität Tübingen, 1993.

[17] Klein, R., and Slusallek, P. Object-oriented framework for curves and surfaces. In *Curves and Surfaces in Computer Vision and Graphics III* (november 1992), J. Warren, Ed., SPIE, pp. 284–295.

[18] Lawson, C. Software for C^1 surface interpolation. In *Mathematical Software III*, J. Rice, Ed. Academic Press, 1977, pp. 161–164.

[19] Lee, D. T., and Schachter, B. J. Two algorithms for constructing a delaunay triangulation. *International Journal Computer and Information Sciences 9*, 3 (1980), 219.

[20] Löhner, R., and Parikh, P. Generation of three dimensional unstructured grids by the advancing front method. *International Journal for Numerical Methods in Fluids 8*, 10 (1988), 1135–1149.

[21] Mäntylä, M. *Solid Modeling*. Computer Science Press, Inc., 1988.

[22] Mitchell, S. A., and Vavasis, S. A. Quality mesh generation in three dimensions. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.* (1992), pp. 212–221.

[23] REQUICHA, A. A. G., AND VOELCKER, H. B. Solid modeling: Current status and research directions. *IEEE Computer Graphics And Applications 3* (Oct. 1983), 25–37.

[24] RIPPA, S. Minimal roughness properties of the delaunay triangulation. *Computer Aided Geometric Design 7*, 6 (1990), 489–498.

[25] RIPPA, S. Adaptive approximation by piecewise linear polynomials on triangulations of subsets of scatterd data. *SIAM Journal Sci. Stat. Comput. 13*, 5 (September 1992), 1123–1141.

[26] SAPIDIS, N. S., AND PERUCCHIO, R. Domain delaunay tetrahedrization of solid models. *Internat. J. Comput. Geom. Appl. 1*, 3 (1991), 299–325.

[27] SHENG, X., AND HIRSCH, B. E. Triangulation of trimmed surfaces in parametric space. *Computer Aided Design 24*, 8 (August 1992), 437–444.

[28] SHIMADA, K., AND GOSSARD, D. Computational methods for physically-based fe mesh generation. In *Human Aspects in Computer Generated Manufactoring*, G. Olling and F. Kimura, Eds. Elsevier Science Publishers B.V. (North Holland), 1992, pp. 411–420.

[29] SZABO, B. Geometric idealizations in finite element computations. *Communications in applied numerical methods 4*, 3 (1988), 393–400.

[30] VIGO, M., AND BRUNET, P. Picewise linear approximation of trimmed surfaces. In *Geometric Modelling*, G. Farin, H. Hagen, and H. Noltemeier, Eds. —, 1993.