Numer Algor (2008) 48:29–47
DOI 10.1007/s11075-008-9188-5

ORIGINAL PAPER

Isotropic umbrella based triangulation of regular parametric surfaces

Victoria Hernández-Mederos · Pedro L. del Ángel · Jorge Estrada-Sarlabous

Received: 21 September 2007 / Accepted: 14 February 2008 /
Published online: 22 April 2008

Abstract In this paper we propose an advancing front method for generating an isotropic triangular mesh on a regular parametric surface. Starting from a point on the surface, the method computes a set of points in the intersection curve between the surface and the sphere centered at that point with a prescribed radius. From this set we select the vertices of a cell composed by triangles approximately equilateral. The mesh grows repeating the described computation with boundary vertices of the cell as starting points. Compared to methods proposed by other authors, the current method may be considered as an improvement, since it is more efficient and flexible. Furthermore, the resulting mesh is closer to being isotropic. Additionally, we obtain a sufficient condition ensuring that a surface triangulation is of Delaunay type.

Keywords Parametric surfaces · Triangular meshes

Mathematics Subject Classifications (2000) 65D17 · 65D07

V. Hernández-Mederos (✉) · J. Estrada-Sarlabous
Instituto de Cibernética, Matemática y Física,
ICIMAF, La Habana, Cuba
e-mail: vicky@icmf.inf.cu

J. Estrada-Sarlabous
e-mail: jestrada@icmf.inf.cu

P. L. del Ángel
Centro de Investigaciones Matemáticas,
CIMAT, Guanajuato, México
e-mail: luis@cimat.mx

1 Introduction

Parametric surfaces are the standard in computer aided design and manufacturing (CAD/CAM) systems. Therefore, a lot of work has been done to obtain good discretizations of parametric surfaces, suitable for visualization [7, 13], for numerical solution of partial differential equations [1, 5, 9, 10, 12] using the finite elements method (FEM) or the boundary elements method (BEM). Triangulation of parametric surfaces is also an essential problem in computer graphics, computational geometry and rapid prototyping. The most common discretization of a surface is a triangular mesh, due to its flexibility for representing complex geometries and also because current graphic hardware and software are tuned to handle triangular meshes. The simplest way of constructing a triangular mesh on a parametric surface consists of generating a (nice) triangulation in parameter space and lifting it to the surface by means of the parametrization. Unfortunately, this triangular mesh on the surface could be very distorted, see Fig. 1 (also Section 4.1), where we have introduced levels of gray tones to visualize shape deformation of triangles (white is an equilateral triangle).

Methods for generating triangular meshes on a surface can be classified in several ways depending on different criteria.

- If we focus on the domain, there are two groups of methods: those computing the mesh in the planar parametric space [3, 5, 7, 12] and those working directly on the 3D surface [1, 9, 10]. Methods in the first class obtain the vertices of the surface mesh mapping a 2D triangular mesh by means of the surface parametrization. These methods have to take into account that the current parametrization usually introduces deformations in length and angles and therefore properties of the 2D mesh are no longer preserved after lifting it on the surface. In other words, well shaped meshes in 2D parametric space could be very twisted after mapping them on the surface. Methods in the second class do not have these problems and can

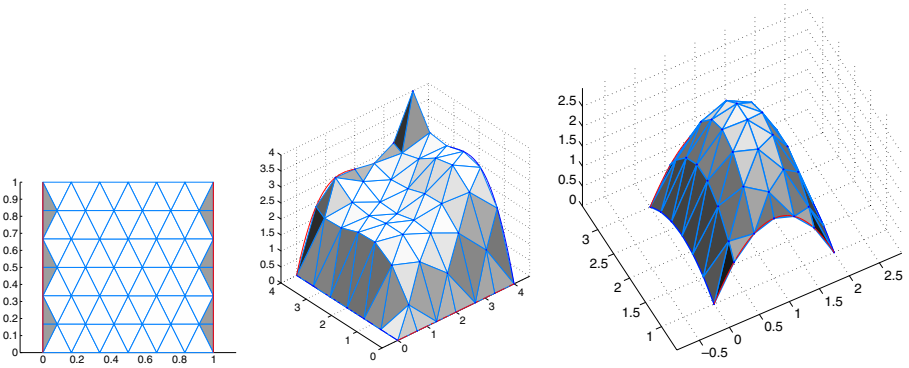

Fig. 1 *Left* a triangulation in parametric space with almost all triangles equilateral, *right* its image using two different parametric surfaces. Observe the deformation in the triangle's shape due to the parametrization

be very useful when surface parametrization is distorting, but usually are more expensive due to computations on the surface.

- Concerning the main technique used by the method there are also two principal classes: methods based on Delaunay triangulations [3, 4, 9, 10] and advancing front methods [1, 5, 12] which start from a cell whose boundary grows as the method progresses.
- Additionally, methods can be classified according to the mesh nature. Some methods generate isotropic meshes [1, 9], in which the size and the shape of triangles are roughly the same, while others [10] produce anisotropic meshes with different density and shape of triangles, depending on the surface curvature. Isotropic meshes are more commonly used in FEM or BEM applications, where numerical algorithms require the triangle shape and size to be as regular as possible since too thin or distorted elements increase the analysis error and slow the solution convergence. Anisotropic meshes are preferred for visualization purposes.

In this paper we propose an advancing front method to generate a triangular mesh on a parametric surface. The method is inspired in the ideas introduced in [1] and the resulting surface mesh is isotropic and therefore well adapted for FEM or BEM applications. Furthermore, the isotropic mesh can be easily refined in order to get a better approximation of the surface, making the method also useful for visualization of the surface. The structure of the paper is the following: in Section 2 we address the previous work, while Section 3 formally presents the problem and the proposed solution. Section 4 describes different strategies to measure and improve the triangulation quality. Section 5 explains how to extend the proposed method to implicit surfaces. Section 6 shows numerical examples and the last section contains concluding remarks.

2 Previous work

In the following we briefly describe the main ideas contained in the previous literature about triangulations on a surface.

In [5, 12] and [3] the advancing front technique is used to generate the triangular mesh, starting with the discretization of the boundary curves and marching from the boundary towards the interior. These methods work on the 2D parametric space and impose a compromise between a nodal density function and the size and quality of the triangles on the surface mesh, based on the local Riemannian metric associated to the surface parametrization. The approach in [5] happens to be relatively expensive, since all distances involved are computed as true curvilinear distances on the surface. On the other extreme, in [12] the estimation of distances between two points on the surface is poor, since it is derived from the application of the local expression for the Riemannian metric on the surface to points which are not necessarily very close. In [3] it is claimed (without proof) that the triangular surface mesh obtained by proposed method is of Delaunay type. The Delaunay triangulation of a set of points on a surface is characterized by the empty "surface

circumcircle" property, see for instance [4]. Given a triangle with vertices on the surface, a "surface circumcircle" is the intersection curve between the surface and the sphere centered in a point on the surface and passing through the vertices. A surface triangulation is of *Delaunay* type if the "circumcircle" associated to each triangle of the mesh does not contain any vertex of the mesh in its interior. Authors of [3] assume that the preimages in parameter space of "surface circumcircles" may be well approximated by ellipses; a very strong hypothesis.

In [9] and [10] the mesh is constructed directly on the surface. A hierarchical subdivision of the surface provides the initial guess for the placement of the mesh vertices or nodes. This mesh is relaxed, assuming that each node is the center of a sphere or bubble in [9] and the center of an ellipsoid or ellipsoidal bubble in [10], under the action of repulsive-attractive forces depending on the relative position of the nodes. The position of the nodes in the relaxed mesh is the solution of a large system of second order non linear ordinary differential equations. The integration process is repeated until it approaches the equilibrium, hence it becomes an expensive procedure. Furthermore, theoretical conditions for the existence of such equilibrium are not given. The problem of how to propose a good initial configuration remains open.

The mesh generated in [9] is isotropic and its edges are constructed connecting the vertices in the parameter space by means of a constrained Delaunay triangulation. Consequently, the node connection is actually decided in parameter space, even when the best node connection in 2D may not be the optimum when the mesh is mapped on the surface. In [10], the mesh is anisotropic and its edges are provided by an anisotropic constrained Delaunay triangulation of the vertices in the 2D parameter space.

Several methods for triangular surface mesh generation are iterative [7–10] and start with an initial mesh (2D or 3D), which it is repositioned as the iteration progresses. Unfortunately, most of these methods lack a formal complexity estimate and convergence analysis. An exception is [7], where a formal proof of the second order of algorithm convergence is included. The main idea of the method proposed in [7] is to obtain a reparametrization of the surface, that behaves approximately like a conformal map in a finite number of points. These points are the vertices of a planar triangulation in the parameter space, whose image by the reparametrization is a nice triangular mesh on the surface. Additionally some of the iterative methods have the disadvantage that the number of vertices must be fixed in advance, instead of depending on the surface area and curvature.

The method we propose in this paper is close to methods in [1] and [6] (originally designed for implicit surfaces) in the sense that all are based on a local tessellation primitive, consisting of a fan of triangles around a point on the surface. In [1] and [6] a regular polygon in the tangent plane to the surface is used to define the fan or umbrella. In [6] vertices of the polygon are projected on the surface, computing the length on the tangent plane and therefore introducing significant errors in regions of high curvature. In [1], for each edge (of size r) of the polygon in the tangent plane at a point P on the

surface, a curve on the surface starting at P and ending in a point Q such that $\|P - Q\| = r$ is calculated.

3 Isotropic meshing

Given a regular parametric surface,

$$F(u, v) = (x(u, v), y(u, v), z(u, v)) \tag{1}$$

with $(u, v) \in \Omega = [0, 1] \times [0, 1] \subset \mathbb{R}^2$ we would like to construct a triangular mesh τ whose vertices are on the surface and such that each triangle is as close as possible to be equilateral. Our method to construct the mesh τ is an advancing front technique inspired in the ideas introduced in [1]. The method starts by selecting a point $F(p_0),\ p_0 \in \Omega$ on the surface and a value r for the length of the edges. Then a triangular cell or umbrella centered at $F(p_0)$ is constructed. Its vertices $F(p_1), ..., F(p_n),\ p_i \in \Omega,\ i = 1, ..., n$ are selected in such a way that:

- All triangles in the cell are as close as possible to be equilateral.
- $\|F(p_i) - F(p_0)\| = r,\ i = 1, ..., r$.

The process is repeated using each neighbor of $F(p_0)$ as the center of a new cell until the boundary curves of the surface $F(u, v)$ are reached.

3.1 Local umbrella

In this section we describe how to construct the umbrella centered at a point $F(p_0)$ on the surface. Since all triangles of the local umbrella centered at $F(p_0)$ must be approximately equilateral, the vertices $F(p_1), ..., F(p_n)$ are on the curve C_S which is defined as the intersection between the surface $F(u, v)$ and the sphere S with radius r and center $F(p_0) = (x_0, y_0, z_0)$

$$S : (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = r^2 \tag{2}$$

We assume that the intersection curve C_S is connected, which can be achieved if the radius r is small enough. Let L be the arc length of the curve C_S. The number of vertices of the cell centered at $F(p_0)$ is computed as $n = \lfloor \frac{L}{r} \rfloor$, where $\lfloor\ \rfloor$ denotes the floor function. In practice, sometimes it is more convenient to construct a cell with $n = \lfloor \frac{L}{r} \rfloor + 1$ vertices, see Section 3.4. With the previous value of n we have to find points $F(p_1), ..., F(p_n)$ on the curve C_S such that

$$\|F(p_i) - F(p_{i+1})\| \approx r, \quad i = 1, ..., n - 1$$

The arc-length of C_S is computed approximately as the arc-length of a polygonal curve defined by N points on C_S with $N >> n$. In Section 3.4 we describe a method that, starting from a large set of N points on a curve, computes a subset of n points, such that the *Euclidean* distance between two consecutive points is approximately the same.

Substituting the coordinate functions (1) of F in (2) we obtain,

$$h(u, v) := (x(u, v) - x_0)^2 + (y(u, v) - y_0)^2 + (z(u, v) - z_0)^2 - r^2 = 0 \quad (3)$$

Observe that (3) is the implicit equation of the curve C_p in the parameter space whose image by $F(u, v)$ is the curve C_S. Let's assume that C_p can be parametrized as $C_p(t) = (u(t), v(t))$. To compute the tangent vector $\frac{dC_p}{dt} = (u_t, v_t)$ to the curve C_p, we derive in (3) and obtain $h_u u_t + h_v v_t = 0$, where h_u, h_v are partial derivatives of the function $h(u, v)$. From this expression we conclude that,

$$u_t = -\lambda(t) h_v \quad (4)$$

$$v_t = \lambda(t) h_u \quad (5)$$

with $\lambda(t)$ a function from $\mathbb{R}$ to $\mathbb{R}$. Observe that the curve C_p can always be parametrized in such a way that $\lambda(t) \equiv 1$ in equations (4) and (5).

Computing h_u and h_v from (3) and substituting in (4) and (5) we obtain then,

$$-u_t = 2(x(u, v) - x_0)x_v + 2(y(u, v) - y_0)y_v + 2(z(u, v) - z_0)z_v \quad (6)$$

$$v_t = 2(x(u, v) - x_0)x_u + 2(y(u, v) - y_0)y_u + 2(z(u, v) - z_0)z_u \quad (7)$$

where $x_u, x_v, y_u, y_v, z_u, z_v$ are partial derivatives of the coordinate functions of $F(u, v)$. The solution of the system of two ordinary differential equations (6) and (7) gives us a parametrization of the curve C_p. Instead of solving this system, we compute a set of points on the implicit curve C_p using Chandler's method, which gives us a large sample of points on C_p. This method requires the tangent vector at the initial point of the curve C_p, which can be easily obtained evaluating the right side of equations (6) and (7).

To generate the local umbrella centered at a point $F(p)$ we must distinguish between 2 cases, see Fig. 2. If p is the first center we generate all the vertices of the local umbrella centered at $F(p)$. Otherwise, we already know some vertices of the umbrella centered at $F(p)$ and we only need to "close" it. We say that a segment of C_S between points $F(p_s)$ and $F(p_e)$ is the *correct* segment, if it does not already contain any vertex of the cell centered at $F(p)$. In that case, we also say that the corresponding segment of C_p between points p_s and p_e is the *correct* segment of C_p. In Fig. 3 the continuous curve is the correct segment of the curve C_p.

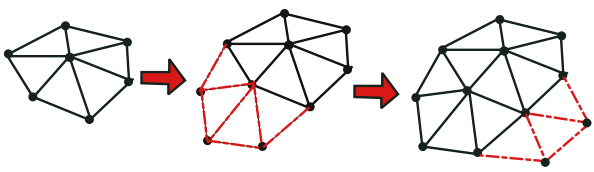

Fig. 2 *Left* first local umbrella, *right* completing two local umbrellas

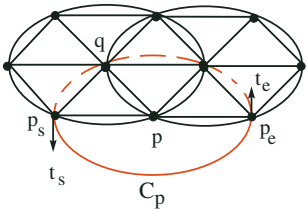

Fig. 3 Choosing the *correct* segment of the curve C_p

To construct the umbrella of the triangles centered at a point $F(p)$ we proceed then as follows. First, we compute an ordered sample $p_1, ..., p_N$ of N points on the *correct* segment of the curve C_p using its implicit equation (3). The value of N has to be large since we assume that the polygonal with vertices at the sample points $F(p_i), i = 1, ..., N$ is a good approximation of the curve C_S. Then we approximate the arc-length L of C_S as the length of the polygonal with vertices $F(p_i), i = 1, ..., N$. From these points we select a subset of n points $F(p_{i_1}), ..., F(p_{i_n})$ such that $\|F(p_{i_j}) - F(p_{i_{j+1}})\| \approx r$, $i = 1, ..., n-1$.

3.2 Algorithm

Now we give a description of the algorithm to construct the isotropic triangular mesh.

Input:

- $F(u, v) = (x(u, v), y(u, v), z(u, v))$, a regular parametric surface defined in Ω.
- $r > 0$ approximate length of the edges.

Output: a triangular mesh $\tau = (V, E)$, with V the set of vertices and E the set of edges, such that the elements of V lie on $F(u, v)$ and the elements of E have length r approximately.

Initialization steps

- Define an empty stack R and two empty sets V and E.
- Choose a point $p_0 \in \Omega$ (for instance, choose p_0 equal to the center of mass of Ω). Compute $F(p_0)$ and push p_0 in R.

Main loop

while $R \neq \emptyset$ do

1. Pop the first point p of the stack R.
2. If p is "very close" to the boundary $\partial\Omega$ of Ω compute a point $F(\widetilde{p})$ close to $F(p)$ with $\widetilde{p} \in \partial\Omega$, set $R := R \setminus \{p\}$, substitute $F(p)$ by $F(\widetilde{p})$ in the set V and p by $\widetilde{p}$.
3. Choose the starting point p_s and the end point p_e of the "correct" segment of C_p.
4. Compute an ordered sample of N points $p_1, ..., p_N$ on the "correct" segment of C_p using its implicit equation (3), see Section 3.3.
5. Compute the points $F(p_1), ..., F(p_N)$ on the "correct" segment of the curve C_S.

6. Approximate the length L of the "correct" segment of C_S by the length of the polygonal curve with vertices at the points $F(p_1), ..., F(p_N)$.
7. From $F(p_1), ..., F(p_N)$ choose an ordered subset of n points $F(p_{i_1}), ..., F(p_{i_n})$ with $F(p_{i_1}) := F(p_s), F(p_{i_n}) := F(p_e), \quad \|F(p_{i_j}) - F(p_{i_{j+1}})\| \approx r$, $j = 1, ..., n-1$, see Section 3.4.
8. Update V including the new vertices $F(p_{i_1}), ..., F(p_{i_n})$. Update E including the new edges $[F(p), F(p_{i_j})]$, $j = 1, ..., n$ and $[F(p_{i_j}), F(p_{i_{j+1}})]$, $j = 1, ..., n-1$.
9. Push $F(p_{i_j})$, $j = 1, ..., n$, in the stack R.

end

Remark

- If $p = p_0$ in step 3 we have to compute all the vertices of the local umbrella centered at $F(p_0)$. Therefore, the correct segment of the curve C_{p_0} is the whole curve and we set $p_s := p_1$ and $p_e := p_1$ where p_1 is an arbitrary point on C_{p_0}.

3.3 Chandler's method

Chandler's method [2] is very efficient to compute a large sample of points p_i, $i = 1, ..., N$ which approximately lie on a given implicit curve $h(u, v) = 0$. Additionally, it has the attractive property that points trace out the curve, i.e. the polygonal with vertices at the points p_i, $i = 1, ..., N$ is a good approximation of the implicit curve. Therefore, its length can be used as an approximation to the arc-length of the curve. Starting from a point p_1 on the curve and the tangent vector at p_1, Chandler's method evaluates the function h at the 8 neighbors of p_1 looking for a change of sign in two function evaluations at midpoints between consecutive pixels. To obtain a large ordered sample of points on the curve C_p (step 4) we implemented Chandler's algorithm taking into account the following aspects.

First, it is necessary to correct the position of the starting point p_s and ending point p_e of the *correct* segment of curve C_p. In fact, if C_p is not the first curve, then $F(p_s)$ and $F(p_e)$ are both vertices of a previously generated local umbrella. In consequence, they are not *exactly* on the sphere with radius r and center $F(p)$ and p_s and p_e are not *exactly* on the curve C_p. Using p_s as initial approximation we adjust its position looking for a point which minimizes the function,

$$g(a) = \alpha\, d_{T_1}(a, C_p) + (1-\alpha)\, \|a - p_s\|, \quad \alpha \in [0, 1]$$

where $d_{T_1}(a, C_p) = \frac{|h(a)|}{\|\nabla h(a)\|}$ is an approximation of the Euclidean distance [11] from the point a to the curve $C_p : h(u, v) = 0$, with $\nabla h(a) = (h_u(a), h_v(a))$. Function $g(a)$ establishes a compromise between distance to the curve C_p and distance to p_s. In our experiments, we used $\alpha = 0.85$. The position of the end point p_e is adjusted similarly.

Let t_s, t_e be the normalized tangent vectors to the curve C_p at p_s and p_e. Assuming that $F(p_s)$ belongs to the local umbrella centered at $F(q)$, we

compute an auxiliary point $\widetilde{F}_{p_s}$, lying on the tangent line to the intersection curve between the surface and the sphere $S_{F(p)}$ of radius r centered at $F(p)$. To select the *correct* segment of C_p we check if $\widetilde{F}_{p_s}$ is outside $S_{F(q)}$. In positive case p_s is the initial point of C_p, which means that starting at p_s in the direction given by t_s we trace out the *correct* segment of the curve C_p. Otherwise, p_e is the initial point and t_e the tangent vector which give us the *correct* segment of C_p.

Recall that the adjusted position of points p_s and p_e are used only to generate the sequence of points on the implicit curve, by means of Chandler method. Nevertheless, the original points $F(p_s)$ and $F(p_e)$ are preserved as vertices of τ.

3.4 Points with uniform distribution

Once we have computed a large ordered sample of points $p_i,\ i = 1, ..., N$ on the implicit curve C_p we can immediately obtain a large ordered sample of points $F(p_i),\ i = 1, ..., N$ on the intersection curve C_S between the surface and the sphere with center $F(p)$ and radius r. In this section we propose an algorithm to select, from $F(p_1), ..., F(p_N)$, a subset of points $F(p_{i_1}), ..., F(p_{i_n})$, with $n < N$ and such that,

$$\|F(p_{i_j}) - F(p_{i_{j+1}})\| \approx r, \quad j = 1, ..., n-1$$

We begin computing $n = \lfloor \frac{L}{r} \rfloor$ and assigning $F(p_{i_1}) = F(p_1)$. Then we calculate the distance from $F(p_{i_1})$ to the next points until we find $j < N$ such that $\|(F(p_1) - F(p_j)\| \leq r$ and $\|(F(p_1) - F(p_{j+1})\| \geq r$. We select $F(p_{i_2}) = F(p_j)$ if the Euclidean distance between $F(p_{i_1})$ and $F(p_j)$ is closer to r than the Euclidean distance between $F(p_{i_1})$ and $F(p_{j+1})$, otherwise we assign $F(p_{i_2}) = F(p_{j+1})$. Starting now with $F(p_{i_2})$, we repeat the process until we have $n-1$ points. Since the length L of the curve C_p is not necessarily a multiple of r and the Euclidean distance between two points in the curve is smaller than the arc- length of the corresponding segment of curve, the distance between $F(p_{i_{n-1}})$ and $F(p_N)$ may be relatively far away from r. To correct this problem we decide if it would be better to include in the subset another point between $F(p_{i_{n-1}})$ and $F(p_N)$.

Finally, to improve the results, we repeat the previous procedure using instead of r some values $\widetilde{r}$ in the interval $[0.9, 1.1]\,r$. For each value of $\widetilde{r}$, we compute the corresponding subset and select as the final subset the one for which the ratio m/M is maximum, where m and M are the minimum and maximum Euclidean distance respectively between two consecutive points in a subset.

3.5 In the proximity of boundary curves

Sometimes, when p is a point close to the boundary of Ω, a segment of the curve C_p is in the exterior of the unit square Ω. We say that p is an *interior* point if it belongs to the interior of the region Ω, otherwise we say that it

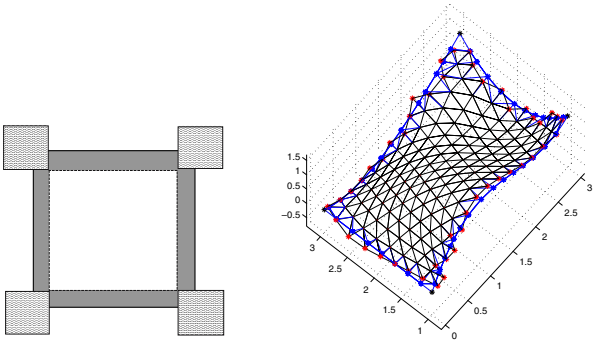

Fig. 4 *Left, dark grey* interior points close to the boundary of Ω, *Wavy* points close to the corners, *right* a surface mesh, vertices close to boundary curves are substituted by boundary points

is an *exterior* point. Any time we pop a point p from the stack R, we check if it is close to one of the corners of Ω, see Fig. 4, Left. If the answer is positive, we compute the projection of $F(p)$ on each of the boundary curves passing through the corresponding corner of $F(u, v)$ and substitute $F(p)$ with the closest projection. If p is an exterior vertex not close to any corner of Ω, we compute the projection $F(q)$ of $F(p)$ on the closest boundary curve and substitute $F(p)$ by $F(q)$. If p is an interior vertex not close to any corner of Ω, but close to the boundary of Ω (p is in the grey zone of Fig. 4), we compute the projection $F(q)$ of $F(p)$ on the closest boundary curve and substitute $F(p)$ by $F(q)$ if and only if $\|F(q) - F(p)\| \leq r/2$.

In any of the previous cases, if an original vertex $F(p)$ is substituted by a point $F(q)$ on the boundary curves, then we check if the Euclidean distance between $F(q)$ and any of the corners $Q_0 = F(0, 0)$, $Q_1 = F(1, 0)$, $Q_2 = F(1, 1)$, $Q_3 = F(0, 1)$ is smaller that $r/5$. In the positive case we substitute $F(q)$ by the corresponding corner. If at the end of the whole process a corner Q_i has not been previously included as vertex of the 3D triangulation, we include it in the list V and include in E the edges of the triangle with vertices Q_i and the closest vertex to it in each boundary curve passing through Q_i.

The basic algorithm described above can only manage open surfaces with the same topology as Ω. In order to generate a triangular mesh on surfaces with different topologies, it is necessary to make a postprocessing for the vertices of the mesh corresponding to parameter values close to the boundary of Ω. In [1] it is described a sewing procedure for managing this case.

4 Triangulation quality

The quality of a 3D triangulation approximating a parametric surface can be measured in several ways. If the triangulation will be used in FEM or BEM applications, then triangle shape and size should be as regular as possible since too thin or distorted elements increase the analysis error and slow the solution convergence. In this sense the "ideal" is a 3D triangulation where all triangles are equilateral. On the other hand, if our main purpose is to render the surface using as approximation the 3D triangulation, then we need a procedure to

refine the mesh in order to reduce the approximation error. Finally, we also study the Delaunay character of the isotropic triangulation proposed here.

4.1 Measuring the quality

Given a triangle with sides of length a, b, c, an indicator of its shape quality, commonly used in the literature [5], is given by,

$$K = \frac{2\sqrt{3}\rho}{\max(a, b, c)} \tag{8}$$

where ρ is the radius of the inscribed circle. The value of ρ can be computed in terms of a, b, c as $\rho = \sqrt{|((s-a)(s-b)(s-c))/s|}$, with $s = (a+b+c)/2$. The constant K measures the proximity of a triangle to be equilateral. In general, $0 \leq K \leq \sqrt{3}$ and K is near to 0 if the length of a side is too small in comparison with the other sides or if the vertices are almost collinear. On the contrary, K is close to $\sqrt{3}$ if one side is too big in comparison with the others. Finally, if $a = b = c$, the triangle is equilateral, then $K = 1$. In the section of numerical experiments we use the value of K to define a gray shading of a triangle in order to visualize its quality.

4.2 Adaptive meshing

Starting from the isotropic mesh that we have introduced in this paper, it is possible to obtain a new mesh suitable for rendering the surface, using the method proposed in [1]. The strategy is the following: the user chooses a value for the length r of the edge representing the smallest feature that one wishes to detect on the surface. Then, we construct the isotropic mesh using the algorithm of Section 3.2 and start the refining process (Fig. 5). Given a bound ε for the distance from an edge to the surface, the refining process computes, for each edge e, the farthest point from the corresponding curve $F(e)$ on the surface and a new vertex is introduced at this point if its distance to e is bigger than ε.

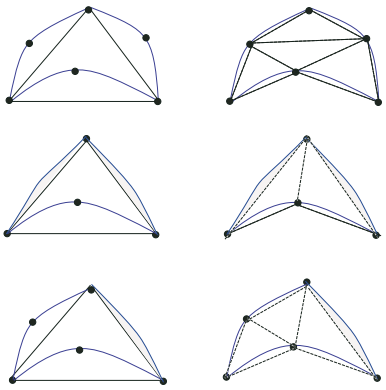

Fig. 5 *Left column* before refining, *right column* after refining. *Top* all edges are split and four new triangles are introduced, *middle* only one edge is split and two triangles are introduced, *bottom* two edges are split and three new triangles are introduced

4.3 Delaunay property

The Delaunay triangulation of a point set, in $\mathbb{R}^2$ or $\mathbb{R}^3$, is very popular due to its nice geometric properties. It is composed of triangles in $\mathbb{R}^2$ and tetrahedra in $\mathbb{R}^3$. A triangulation with vertices on a surface corresponds to an intermediate case, since the vertices are 3D points but the surface is a 2D topological object. As a consequence, it is necessary to adapt the concept of Delaunay triangulation for this special case. We use the definition given in [4].

Definition 1 [4] Given three vertices on a curved surface, consider the infinite set of spheres through the three vertices. The centers of all the spheres lie on a single line. We choose the sphere whose center is on the surface and define the *circumcircle* of the three vertices to be the set of points where this sphere intersects the surface.

Definition 2 A surface triangulation is of *Delaunay* type if the circumcircle associated to each triangle of the mesh does not contain any vertex of the mesh in its interior.

In this section we obtain a sufficient condition ensuring that a surface triangulation is of Delaunay type. As a particular case, we prove that any isotropic triangulation (with vertices on a surface) which provides a good approximation of a surface is a Delaunay triangulation.

Definition 3 Given a triangle T with vertices on F, the parameter values of its vertices define a triangle $\mathfrak{T}$ in parameter space Ω. The distance $d(T, F)$ from F to T is defined as $max_p \{d(F(p), T) \,/\, p \in \mathfrak{T}\}$, where $d(F(p), T) = \|F(p) - q\|$, with $q \in T$ and such that $F(p) - q$ is parallel to the normal vector of T. Further, for any triangulation τ on F we call $max_T \{d(T, F) \,/\, T \in \tau\}$ the distance between τ and F.

Theorem 1 *Let α, β, r be nonnegative real numbers and let $\tau = (V, E)$ be a triangulation with V the set of vertices and E the set of edges of τ such that,*

1. *If $v \in V$ then it is on a surface F*
2. *If $e \in E$ then $(1-\beta)r \leq \|e\| \leq (1+\beta)r$, $r > 0$*
3. *$\|v_i - v_j\| > r$ for all $v_i, v_j \in V$, $i \neq j$ such that $\overline{v_i v_j} \notin E$*
4. *The maximum distance ε between F and the triangulation τ is not greater than $\alpha\, r$*

If α and β are such that the following inequality holds

$$2\left(\gamma + \alpha^2 + \alpha\sqrt{\gamma + \alpha^2}\right) < (1-\beta)^2 \tag{9}$$

where $\gamma = \frac{(1+\beta)^2}{3}$, then τ is a Delaunay triangulation on the surface F.

Proof Let's consider a triangle $T \in \tau$. We must show that the sphere passing through the vertices of T and whose center is on F does not contain any vertex

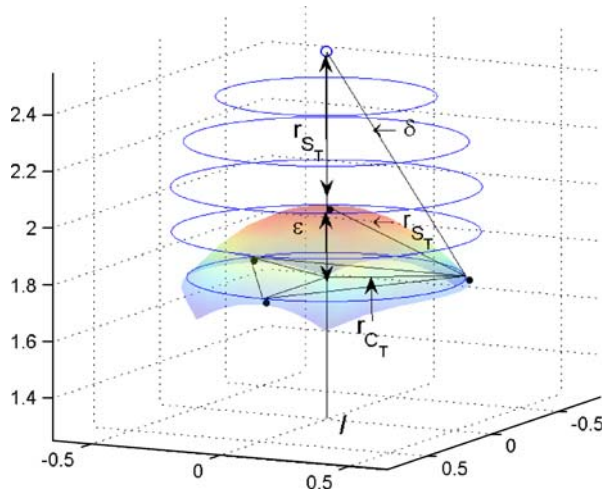

Fig. 6 Circumsphere associated to a triangle on τ. Black points are on the surface

$v \in V$ in its interior. From now on we refer to this sphere as the circumsphere associated to T and denote it by S_T. Observe that the center of S_T lies on the line l passing through the center C_T of the circumscribed circle, with the direction of the normal to T. Let's denote by r_{C_T} the circumradius of T. From the hypothesis (2), the maximum value for r_{C_T} is obtained when T is an equilateral triangle with edges of length $(1+\beta)r$. In that case, r_{C_T} would have the value $r_{C_T} = (1+\beta)r\sqrt{3}/3$. Therefore in general,

$$r_{C_T}^2 \leq r^2\gamma \tag{10}$$

On the other hand (see Fig. 6), the radius r_{S_T} of the circumsphere S_T satisfies,

$$r_{S_T}^2 = \varepsilon^2 + (r_{C_T})^2 \leq \varepsilon^2 + r^2\gamma \tag{11}$$

Assume that the circumsphere S_T contains a vertex v of τ not belonging to T. Then, the minimum distance δ from v to a vertex of T is maximal when v is the intersection point of the line l with S_T which is further away from T. In that case, $\delta^2 = (r_{S_T} + \varepsilon)^2 + (r_{C_T})^2$ and from (10) and (11) we get,

$$\delta^2 \leq \left(\sqrt{\varepsilon^2 + \gamma r^2} + \varepsilon\right)^2 + \gamma r^2 \tag{12}$$

Since by hypothesis (4) $\varepsilon \leq \alpha r$, from (12) we obtain,

$$\delta^2 \leq \left(\sqrt{(\alpha r)^2 + \gamma r^2} + \alpha r\right)^2 + \gamma r^2 = r^2\left[\left(\sqrt{\alpha^2+\gamma} + \alpha\right)^2 + \gamma\right] \tag{13}$$

From (9) the following inequality holds, $\left(\sqrt{\alpha^2+\gamma} + \alpha\right)^2 + \gamma < (1-\beta)^2$. Therefore, in (13) the expression in brackets is smaller than $(1-\beta)^2$ and we get, $\delta < (1-\beta)r < r$, in contradiction either to (2) or to (3). Hence, no vertex of τ different from the vertices of T may be in the circumsphere S_T. □

Remark

1. Assuming that $\alpha < 0.2$ it is straightforward, but cumbersome, to show that the inequality (9) is equivalent to $((1+\beta)^2 - 12\beta) > \alpha\sqrt{24(1+\beta)^2 - 144\beta}$.
2. Observe that as α decreases, i.e. as the triangulation better approximates the surface, the interval for the edge length grows preserving the Delaunay property.

Two interesting cases may be obtained from the previous Theorem. They correspond to $\beta = 0$ (Corollary 1) and $\alpha = 0$ (Corollary 2).

Corollary 1 *Let $\tau = (V, E)$ be a triangulation with V the set of vertices and E the set of edges of τ such that,*

1. *If $v \in V$ then it is on a surface F*
2. *All triangles $T_i \in \tau$ are equilateral with edge length r*
3. *$\|v_i - v_j\| \geq r$ for all $v_i, v_j \in V, i \neq j$*
4. *The maximum distance ε between F and the triangulation τ is smaller than $r/5$*

then τ is a Delaunay triangulation on the surface F.

Corollary 2 *Let $\tau = (V, E)$ be a planar triangulation with V the set of vertices and E the set of edges of τ, such that if $e \in E$ then $0.9r \leq \|e\| \leq 1.1r$. Then τ is a (planar) Delaunay triangulation.*

5 The implicit case

With some few adaptations, the method we presented above to generate an isotropic triangular mesh on a regular parametric surface may be tuned for regular implicit surfaces.

Given an implicitly defined regular surface $F : F(x, y, z) = 0$ and a point $P_0 = (x_0, y_0, z_0)$ on F, for a radius value r sufficiently small, the sphere S with center P_0 and radius r intersects F in a closed and connected curve C_S.

Substituting the parametric equation in $(u, v) \in [0, 1)$ of the sphere S in the implicit equation of F, we obtain the implicit equation of a curve C_p in the (u, v)-plane,

$$C_p : G(u, v) = 0$$

with $G(u, v) = F(rsin(\pi u)cos(2\pi v) + x_0,\ rsin(\pi u)sin(2\pi v) + y_0,\ rcos(\pi u) + z_0))$.

It is not computationally expensive to use Chandler's method to track the implicit curve C_p in the (u, v)-plane. Given a dense ordered set of points on an arc of C_s, the approximate computation of its arc-length, the generation of a subset of n points on C_S such that the Euclidean distance between two consecutive points is approximately the same and the choice of the correct segment of the curve C_p, described in Sections 3.1–3.4 for the parametric case,

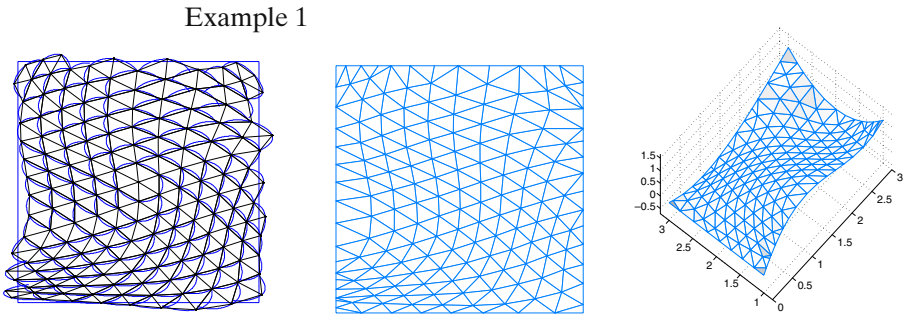

Fig. 7 $F(u, v)$ is a tensor product Bezier surface of degree 2×3, $r = 0.25$, the surface triangulation is Delaunay

may be adapted in a straightforward way to the implicit case, just lifting the points in the (u, v)- plane on the 3D implicit surface F. On the other hand, if we know the cartesian coordinates (x_j, y_j, z_j) of a point P_j on C_S, it is easy to compute its coordinates (u_j, v_j).

The algorithm to compute the isotropic triangular mesh on a parametric surface given in Section 3.2 may be used for regular implicit surfaces taking into account the following facts. Now the stack R consists of points P on the 3D surface F, instead of points p in $[0, 1] \times [0, 1]$. Let us assume that the boundary of the implicit surface F is a closed curve, defined by means of a chain of implicitly defined plane curve segments (arcs), such that two consecutive arcs have only a common point (called corner). If a vertex P from the stack R is close to the boundary of F, we may proceed as in Section 3.5 for the parametric case. Observe that since each arc is a segment of an implicit plane curve, by means of Chandler's algorithm it is possible to compute a dense sample of 3D points representative of each arc and to use this sample to get good approximations for the closest point to a given point P in any arc of the boundary. No hypothesis of Section 4 is based on the assumption that the surface F is represented by a parametrization. Therefore, it is possible to measure the triangulation quality of the proposed algorithm for implicitly

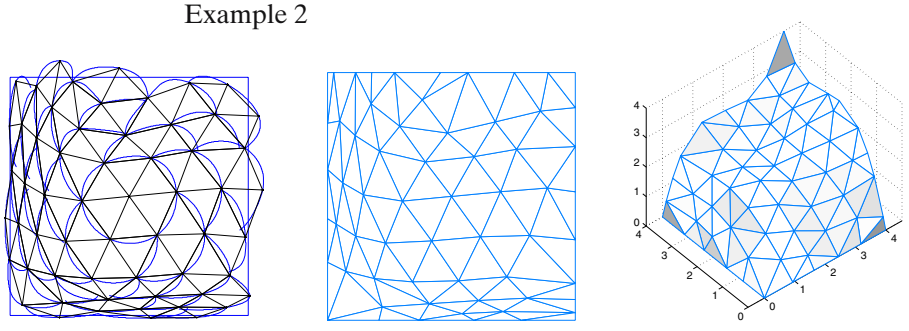

Fig. 8 $F(u, v)$ is a tensor product Bezier surface of degree 4×4, $r = 0.8$, the surface triangulation is Delaunay

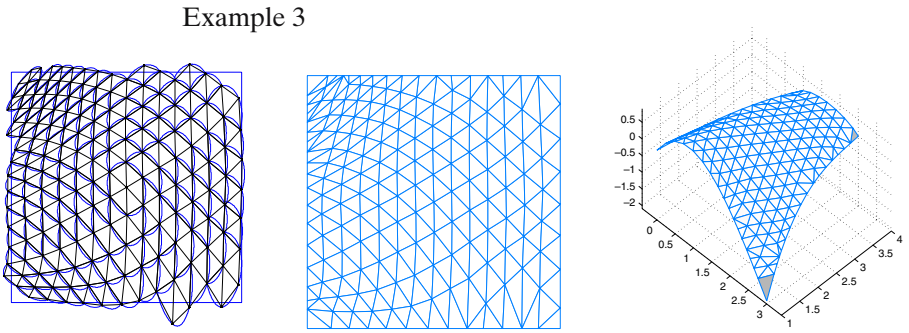

Fig. 9 $F(u, v)$ is a tensor product Bezier surface of degree 3×3, $r = 0.25$, the surface triangulation is not Delaunay

defined surfaces and we may also refine the isotropic triangulation obtained in this way to obtain a new mesh suitable to render a regular implicit surface.

6 Numerical examples

The proposed algorithm to construct an isotropic triangular mesh on a regular parametric surface has been implemented in *Matlab*. We illustrate its performance using different tensor product Bezier surfaces (Figs. 7, 8, 9, and 10). Figures in examples 1–4 show,

- *On the left* the curves C_p and the triangulation in Ω obtained after selecting points with uniform distribution on the surface curves $F(C_p)$.
- *In the middle* the final triangulation in the parameter space Ω obtained after processing boundary curves.
- *On the right* the final triangulation on the surface. We have used levels of gray to visualize shape deformations given by (8) (white is an equilateral triangle, $K = 1$)

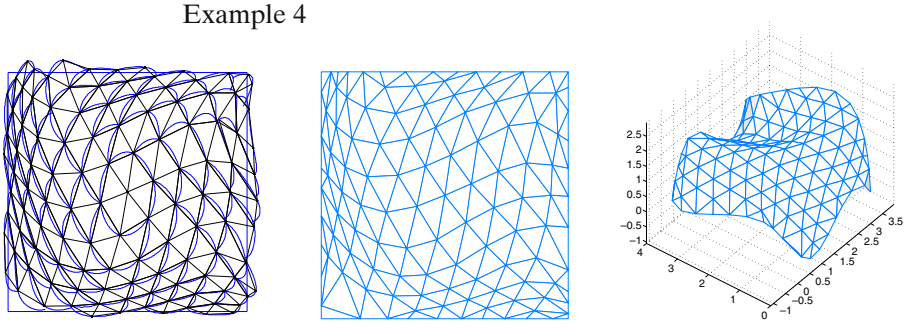

Fig. 10 $F(u, v)$ is a tensor product Bezier surface of degree 4×3, $r = 0.5$, the surface triangulation is Delaunay

Table 1 Performance of the algorithm in the examples

Example	n_t	n_v	r	κ	σ_K^2	μ	$\sigma_{\|e\|}^2$	Error
1 D	286	170	0.25	0.959560	0.002314	0.257517	0.001240	0.030284
2 D	95	62	0.8	0.930312	0.007162	0.825440	0.019588	0.333319
3	298	178	0.25	0.970716	0.001519	0.254809	0.001399	0.019717
4 D	195	120	0.5	0.959705	0.003219	0.499852	0.005314	0.165782

To certify the Delaunay property we check all the triangles, including the ones which have a vertex on the boundary curves (usually the more distorted). Excluding these triangles from the analysis all the triangulations in the examples are of Delaunay type.

In Table 1 we summarize the quality of the triangulations obtained in the examples. A letter "D" besides the example number means that it is a Delaunay triangulation. For each triangulation we include the number n_t of triangles, the number n_v of vertices, the radius r and the following measures of the quality:

- The closeness to isotropy, $\kappa = \frac{1}{n_t}\sum_{i=1}^{n_t} K_i$
- The standard deviation of K, $\sigma_K^2 = \frac{1}{n_t-1}\sum_{i=1}^{n_t}(K_i - \kappa)^2$
- The mean of the edge lengths, $\mu = \frac{1}{m}\sum_{i=1}^{m}\|e_i\|$, where m is the number of edges
- The standard deviation of $\|e\|$, $\sigma_{\|e\|}^2 = \frac{1}{m-1}\sum_{i=1}^{m}(\|e_i\| - \mu)^2$
- The maximum approximation error on the edges, $error = \max_{e\in E} d(e, F)$, where $d(e, F)$ is the distance between the edge e and the corresponding curve $F(e)$ on F

Figure in example 5 shows the refining process described in 4.2. We start constructing the isotropic triangulation for a tensor product Bezier surface of degree 4×4 using $r = 0.8$. The isotropic mesh is refined twice with $\varepsilon = 0.01$ (Fig. 11). All triangles are shaded in grey tones to visualize the approximation error, computed as the normalized mean of the error associated to the edges. We use as normalization factor the maximum error of the triangulations, which is obviously attained for the initial one.

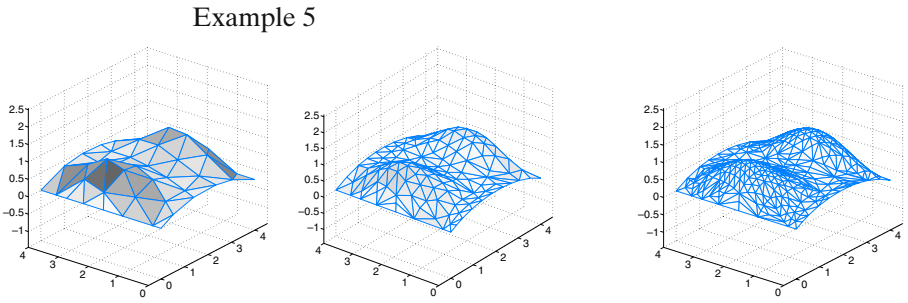

Fig. 11 Refining two times the initial isotropic triangulation

7 Concluding remarks

In this paper we have proposed a method to construct an isotropic triangulation on a regular parametric surface. Additionally, we obtain a sufficient condition ensuring that a surface triangulation is of Delaunay type.

In comparison with other methods reported on the literature our method has the following advantages:

- **Isotropy**: The resulting triangulation is very close to be isotropic as it is shown in the previous numerical examples, where we have used the shape quality indicator (8) to measure the proximity of a triangle to be equilateral. Other authors do not report the behavior of this indicator for their approaches, therefore we cannot make a comparison in this sense.
- **Flexibility**: The resulting isotropic mesh can be refined to obtain an adaptive triangulation suitable for surface visualization. With few modifications the proposed method can be easily used to generate an isotropic triangulation on implicitly defined surfaces. We are currently working on an extension of the method to generate a surface triangulation depending on a nodal density function in order to obtain anisotropic meshes with variable density and size of triangles.
- **Efficiency**: The computational cost of our method is lower than the cost of method [1]. A precise comparison is difficult since authors of [1] do not give details about the implementation of their method. Nevertheless, we can take under consideration the following aspects.
  Let us refer to the method [1] as MA and to our method as MH. Now we show that *once we know the number of vertices of the umbrella* centered at a point P on the surface, the computational cost of MA and MH is similar. First, we recall that for each vertex Q_i of the umbrella centered at P, MA computes a curve S_i on the surface starting at P and ending in Q_i, while MH computes the section C_i of the front curve C_S joining Q_i and Q_{i+1}. For a generic surface, we may assume that if r is small, then curves S_i and C_i have approximately the same geometric behavior (arc-length, bending energy, etc.). To generate a sample of points on the curve S_i, MA uses the adaptive step-size Runge–Kutta method to solve a system of 2 ordinary differential equations (*ode*). On the other hand, MA computes a sample of points on the curve C_i using Chandler's method. The number of points computed by Runge–Kutta method on S_i, could be in general smaller than the number of points obtained by Chandler's on C_i. Nevertheless, the computation of each point on S_i using Runge–Kutta method, requires 6 evaluations for each function in the right hand side of the *ode*, in total 12 evaluations. In comparison, the computation of each point on C_i using Chandler's method requires, in the worse case, 6 evaluations of the function h given by (3). In consequence, the computational cost of both methods, *once we know the number of vertices of the umbrella* is similar.
  But, in order to obtain the number of vertices of an umbrella, it is necessary to estimate the arc-length of the front curve C_S. Since MH is based on

Chandler method to compute a set of points on C_S, these points can be used not only to estimate the arc-length of C_S, but also to select vertices of the umbrella. On the other hand, to obtain an approximation of the arc-length of C_S in MA it is necessary to perform additional computations. In fact, the arc-length of C_S in MA is approximated by the length of a polygonal, whose vertices are the end points of curves (on the surface) starting at P. In order to get a reasonable approximation, the number of curves starting at P has to be relatively big and the computation of each of these curves requires the solution of a system of two *ode*. Therefore, many **extra** *ode* systems have to be solved in MA. Summarizing, the difference in computational cost of MA and MH is dominated by the cost of the solution of many **extra** *ode* systems in MA to estimate the arc-length of the front curve of each umbrella.

Acknowledgements We would like to thank Gert Wollny for his careful reading and suggestions on the first version of this paper. The last version of this paper was written during the stay of the first author at the Mathematics Section of the Abdus Salam ICTP. We want to thank Prof. D.T. Lê for his generous invitation and ICTP for the support. Finally, we wish to acknowledge the anonymous referees for their valuable comments and suggestions.

References

1. Attene, M., Falcidieno, B., Spagnuolo, M., Wyvill, G.: A mapping independent primitive for the triangulation of parametric surfaces. Graph. Models **65**, 260–273 (2003)
2. Chandler, R.: A tracking algorithm for implicitly defined curves. IEEE Graph. Appl. **8**, 83–89 (1988)
3. Chen, H., Bishop, J.: Delaunay triangulations for curved surfaces. In: Proceedings of 6th International Meshing Roundtable, Utah, pp. 115–127, 13–15 Oct 1997
4. Chew, L.: Guaranteed quality mesh generation for curved surfaces. In: Proceedings of 9th Annual symposium on Computational Geometry, pp. 274–280. ACM Press, New York (1993)
5. Cuillere, J.C.: An adaptive method for the automatic triangulation of 3D parametric surfaces. Comput. Aided Geom. Des. **30**, 139–149 (1998)
6. Hartmann, E.: A marching method for the triangulation of surfaces. The Vis. Comput. **14**, 95–108 (1998)
7. Hernández, V., Estrada, J., León, D.: Generating a nice triangular mesh on a regular parametric surface. Computing **27**, 225–235 (2007)
8. Li, S.Z.: Adaptive sampling and mesh generation. Comput. Aided Des. **27**, 235–240 (1995)
9. Shimada, K., Gossard, D.: Automatic triangular mesh generation of trimmed parametric surfaces for finite element analysis. Comput. Aided Des. **15**, 199–222 (1998)
10. Shimada, K., Yamada, A., Takayuki, I.: Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. Int. J. Comput. Geom. Appl. **10**, 400–424 (2000)
11. Taubin, G.: Distance approximations for rasterizing implicit curves. ACM Trans. Graph. **13**, 3–42 (1994)
12. Tristano, J.R., Owen, S.J., Canan, S.A.: Advancing front surface mesh generation in parametric space using a Riemannian surface definition. In: Proceedings of 7th International Meshing Roundtable, Michigan, pp. 429–445, 26–28 Oct 1998
13. Velho, L., Henrique de Figuereido, L.: Optimal adaptive polygonal approximation of parametric surfaces. In: Proceedings SIBGRAPI'96, pp. 127–133, Oct 1996