Mesh Refinement over Triangulated Surfaces

Manolo Castro-Diaz

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Mesh Refinement over Triangulated Surfaces

M.J. Castro Díaz

N° 2462

October 1994

PROGRAMME 6

ISSN 0249-6399

INRIA
ROCQUENCOURT

Mesh Refinement over Triangulated Surfaces

M.J. Castro Díaz*

Programme 6 — Calcul scientifique, modélisation et logiciel numérique
Projet MENUSIN

Rapport de recherche n° 2462 — October 1994 — 40 pages

Abstract: We consider an implementation of Farin's algorithm, with some modifications, for 'C^1' Bézier interpolation of a surface known only by an arbitrary triangulation. This algorithm was originally developed as a tool for computer aided design software, but it can be useful for visualization of Finite Element software outputs in numerical analysis and in remeshing problems of a given surface with unknown parametrization. The method is implemented in C++ with automatic mesh refinement in mind.

Key-words: Surface Interpolation, Mesh Generation, Surface Mesh Generation, Finite Elements.

(Résumé : tsvp)

*Dpto. de Análisis Matemático, Universidad de Málaga, Campus Universitario de Teatinos s/n, 29080 Málaga.

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
Téléphone : (33 1) 39 63 55 11 – Télécopie : (33 1) 39 63 53 30

Raffinement de Maillages Triangulaires Surfaciques

Résumé : Nous considérons ici une implémentation de l'algorithme de Farin modifié, pour l'interpolation Bézier "C^1" des surfaces déterminées par une triangulation arbitraire. Cet algorithme fut originellement conçut comme un utilitaire pour des logicieles de CAO, mais se révèle utile aussi pour la visualisation des résultats des logiciels d'éléments finis et les problèmes de remaillage de surfaces à paramétrisation inconnue. La programmation utilise le langage C++. Le but final est de parvenir à adapter automatiquement des maillages.

Mots-clé : Interpolation de Surfaces, Génération des Maillages, Génération de Maillages Surfaciques, Éléments Finis.

Contents

List of Figures

1 Introduction

We know that some tools of CAD can be useful in numerical analysis [9] for visual display of results by computer graphics. If we want to obtain smoother display of results, we have to interpolate them by a C^1 function. But, here, we consider the following problem that appears in remeshing an arbitrary surface with unknown parametrization.

Suppose we are given a triangulation of a surface S, S_0. This triangulation defines a polygonal surface and we want to remesh it, adding, suppressing and moving their points. In these three processes we have to position points (vertex) over the given surface. We can use the continuous surface S_0 to position the points, but is it possible to do it better? Can we obtain 'a better definition' of the initial surface S using C^1 interpolation over S_0? We find that Farin's algorithm [7] for 'C^1'-linking of Bézier surface patches can be adopted for this task.

Farin's algorithm [7], which has been developed primarily to obtain 'C^1' surfaces by connecting triangular Bézier surface patches, can be applied in numerical analysis and particularly, in our case, in a remeshing problem. Even though, there are no new ideas, the contribution of this work is the implementation of Farin's algorithm in C++ and its use in a remeshing process. This algorithm has been incorporated in the package PPMSH at INRIA.

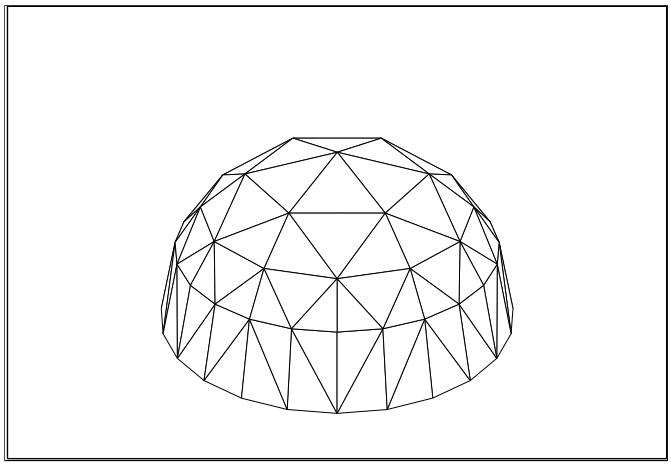

Figure 1.1: Triangulation of an hemisphere

Our problem can be stated in the following way:

Suppose we are given a triangulation of a surface S (see fig.(1.1)) in $\mathbb{R}^3$; that is, we are given

1. a set $\{s_i\}_{1\leq i\leq N_v}$, of points in $\mathbb{R}^3$ lying on S,
2. an array of triangles $\{K_j\}_{1\leq j\leq N_t}$, forming a conforming mesh [4] of the above points, such that, the surface S_0 obtained by linear interpolation over the triangles

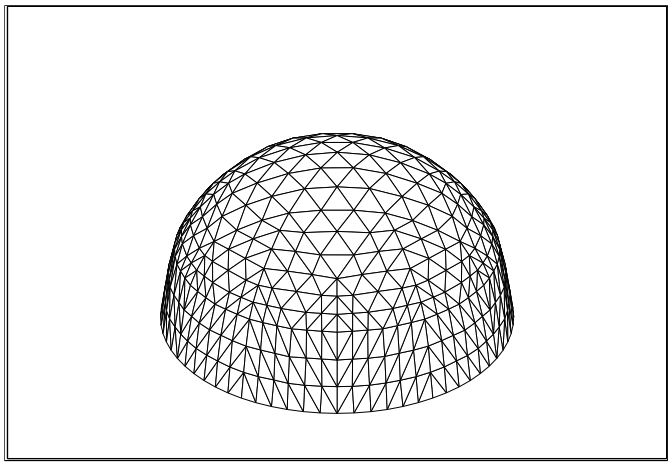

Figure 1.2: The same hemisphere recovered by a 'C^1' interpolation over the initial triangulation

$$S_0 = \bigcup_{j=1}^{N_t} \{ \sum_{i=1}^{3} \alpha_i s_j^i \ : \ \alpha_i \geq 0, \ \alpha_1 + \alpha_2 + \alpha_3 = 1 \}, \tag{1.1}$$

is an approximation to S, where $s_j^i, \ 1 \leq i \leq 3$, are the vertices of the triangle K_j.

The surface S_0 is only C^0, and not necessarily 'C^1' (see definition (5)) . Now, we can state the following problem:

Find another interpolation, $\mathbf{S_h}$*, so that* S_h *shall be '*$\mathbf{C^1}$*', easy to construct and as close to* $\mathbf{S_0}$ *as possible.*

Remark 1 *Fist of all, we remark that this problem, for a general surface, can not be reduced to the problem of* C^1 *interpolation for a surface of the form* $z = f(x, y)$ *over a triangulated* $2D$ *domain knowing the values of* f *at the vertices of the triangulation. Therefore reduced HCT [11] interpolation seems difficult to extend to our case.*

2 Mathematical background

In this section we give some definitions and theoretical results that we are going to use to develop the numerical algorithm presented in the next section.

2.1 Bézier surfaces

Definition 1 *The I-th Bernstein polynomial of degree n in two variables is defined by:*

$$B_I^n(\mathbf{u}) = \frac{n!}{i_1!i_2!i_3!} u_1^{i_1} u_2^{i_2} u_3^{i_3} \quad \mathbf{u} = (u_1, u_2, u_3), \tag{2.1}$$

with

$$I = (i_1, i_2, i_3),\ n = |I| = i_1 + i_2 + i_3,\ u_1 + u_2 + u_3 = 1. \tag{2.2}$$

We define $B_I^n(\mathbf{u}) = 0$ if some of the (i_1, i_2, i_3) are negative.

Remark 2 *The variable* $\mathbf{u}$ *can be regarded as barycentric coordinates of a point p_0 with respect to a triangle K.*

Bernstein polynomials satisfy the following recursion:

Proposition 1 *Let $B_I^n(\mathbf{u})$ be the I-th Bernstein polynomial of degree n in two variables, then*

$$B_I^n(\mathbf{u}) = u_1 B_{I-e_1}^{n-1}(\mathbf{u}) + u_2 B_{I-e_2}^{n-1}(\mathbf{u}) + u_3 B_{I-e_3}^{n-1}(\mathbf{u}), \quad |I| = n, \tag{2.3}$$

where $e_1 = (1,0,0),\ e_2 = (0,1,0)$ and $e_3 = (0,0,1)$.

Proof:

Let $E(\mathbf{u}) = u_1 B_{I-e_1}^{n-1}(\mathbf{u}) + u_2 B_{I-e_2}^{n-1}(\mathbf{u}) + u_3 B_{I-e_3}^{n-1}(\mathbf{u})$. We have:

$$\begin{aligned} E(\mathbf{u}) &= \left[\frac{(n-1)!}{(i_1-1)!i_2!i_3!} + \frac{(n-1)!}{i_1!(i_2-1)!i_3!} + \frac{(n-1)!}{i_1!i_2!(i_3-1)!} \right] u_1^{i_1} u_2^{i_2} u_3^{i_3} \\ &= \left[\frac{i_1}{n} + \frac{i_2}{n} + \frac{i_3}{n} \right] B_I^n(\mathbf{u}) \\ &= B_I^n(\mathbf{u}). \end{aligned}$$

A simple exercise of calculus gives that the partials ∂^J of the Bernstein polynomials are:

$$\partial^J B_I^n(\mathbf{u}) = \frac{\partial^{|J|}}{\partial u_1^{j_1} \partial u_2^{j_2} \partial u_3^{j_3}} B_I^n(\mathbf{u}) = \frac{n!}{(n-r)!} B_{I-J}^{n-r}(\mathbf{u}),\ |J| = r. \tag{2.4}$$

Definition 2 *We define a triangular Bézier surface patch of order n with control points $\{b_I\}_{|I|=n} \in \mathbb{R}^3$ by:*

$$S_b = \{S(\mathbf{u}) = \sum_{|I|=n} b_I B_I^n(\mathbf{u}) : \quad u_i \geq 0, \quad \sum_{i=1,\cdots,3} u_i = 1\}. \tag{2.5}$$

The surface patch S_b can be regarded as a mapping

$$S\colon K \longrightarrow \mathbb{R}^3, \quad \mathbf{u} \in K \mapsto S(\mathbf{u}) \in \mathbb{R}^3,$$

from a triangle K, with $\mathbf{u}$ barycentric coordinates of a point $p \in K$, into $\mathbb{R}^3$.

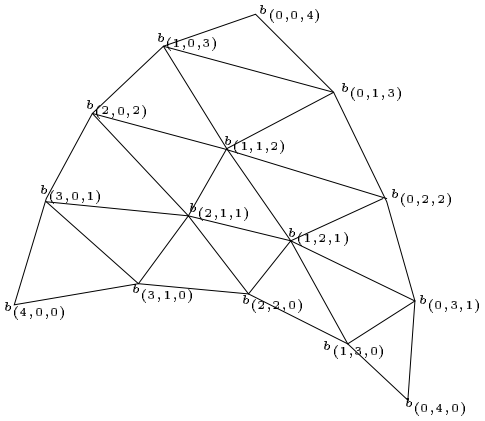

Figure 2.1: Control points for Bézier surface patch of order $n = 4$.

Remark 3 *The three control points $b_{(n,0,0)}$, $b_{(0,n,0)}$ and $b_{(0,0,n)}$ (see fig. (2.1)) are called the vertices of the patch S_b. We also say that S_b is a triangular surface patch over the triangle K with these three points as the vertices.*

Remark 4 *A surface patch S_b passes through the vertices and not necessarily through the other control points.*

Definition 3 *A surface defined as the union of triangular Bézier surface patches over a given mesh is called a Bézier surface over the mesh. If all the surface patches are of order n, then the Bézier surface is said to be of order n.*

The *boundary of a triangular Bézier patch* is defined by:

$$\bigcup_{i=1}^{3} \{S(\mathbf{u}) : u_i = 0, \quad u_j \geq 0, \quad \sum u_j = 1\}. \tag{2.6}$$

So k-th boundary edge, for $k = 1, \cdots, 3$, is a Bézier curve of order n and it depends on the control points b_I with $i_K = 0$ (see fig. (2.1)):

$$\sum_{|I|=n, i_k=0} b_I B_I^n(\mathbf{u})_{|_{u_k=0,\ u_i\geq 0,\ u_1+u_2+u_3=1}}. \tag{2.7}$$

This curve is patch-independent because it is defined by the control points of the k-th boundary edge.

So, if our interest is to obtain a continuous approximation of our surface, we only impose that the control points of two adjacent patches must be equal.

Definition 4 (de Casteljau algorithm) *Given a triangular array of points $b_I^* \in \mathbb{R}^3$ with $|I| = n$ and a point p with barycentric coordinates $\mathbf{u}$, we define:*

$$b_J^r(\mathbf{u}) = u_1 b_{J+e_1}^{r-1}(\mathbf{u}) + u_2 b_{J+e2}^{r-1}(\mathbf{u}) + u_3 b_{J+e3}^{r-1}(\mathbf{u}), \tag{2.8}$$

where $r = 1, \ldots, n$, $|J| = n - r$, $e_1 = (1, 0, 0)$, $e_2 = (0, 1, 0)$, $e_3 = (0, 0, 1)$ and $b_J^0 = b_J^*$.

Proposition 2 *Let b_J^r the intermediate points of the Casteljau algorithm, then they can be expressed in term of Bernstein polynomials as:*

$$b_J^r = \sum_{|I|=r} b_{J+I}^* B_I^r, \quad |J| = n - r. \tag{2.9}$$

Proof:

We are going to use induction and the recursive definition of Bernstein polynomials (see (2.3)).

Case $r = 1$:

$$\begin{aligned} b_J^1(\mathbf{u}) &= u_1 b_{J+e1}^0(\mathbf{u}) + u_2 b_{J+e2}^0(\mathbf{u}) + u_3 b_{J+e3}^0(\mathbf{u}) \\ &= u_1 b_{J+e_1}^* + u_2 b_{J+e_2}^* + u_3 b_{J+e_3}^* \\ &= \sum_{|I|=1} b_{J+I}^* B_I^1(\mathbf{u}). \end{aligned}$$

Now, we suppose that the formula is true for $1, \cdots, r-1$ and we obtain:

$$\begin{aligned} b_J^r &= u_1 b_{J+e_1}^{r-1} + u_2 b_{J+e_2}^{r-1} + u_3 b_{J+e_3}^{r-1} \\ &= u_1 \sum_{|I|=r-1} b_{J+e1+I}^* B_I^{r-1} + u_2 \sum_{|I|=r-1} b_{J+e2+I}^* B_I^{r-1} + u_3 \sum_{|I|=r-1} b_{J+e3+I}^* B_I^{r-1} \end{aligned}$$

$$= u_1 \sum_{|I|=r} b^*_{J+I} B^{r-1}_{I-e1} + u_2 \sum_{|I|=r} b^*_{J+I} B^{r-1}_{I-e2} + u_3 \sum_{|I|=r} b^*_{J+I} B^{r-1}_{I-e3}$$

$$= \sum_{|I|=r} b^*_{J+I} B^r_I.$$

Remark 5 *Setting $r = n$ in the equation (2.9), we obtain*

$$S(\mathbf{u}) = b^n_{\mathbf{0}}(\mathbf{u}) = \sum_{|I|=n} b^*_I B^n_I(\mathbf{u}). \tag{2.10}$$

Remark 6 *We can generalize (2.10) as:*

$$S(\mathbf{u}) = \sum_{|J|=n-r} b^r_J(\mathbf{u}) B^{n-r}_J(\mathbf{u}), \quad 0 \le r \le n. \tag{2.11}$$

We know that partial derivatives are not an adequate tool when dealing with triangular patches. We will use the following generalization of the C^1 regularity definition.

Definition 5 *[5] (Visually continuous). Let Φ and φ be two surface patches that have a common boundary curve Γ, and let $\Gamma'(v)$ denote its tangent vector at point $\Gamma(v)$. Let $D^1_{\mathbf{d_1}}\Phi(v)$ denote a cross-boundary derivative of Φ at $\Gamma(v)$, i.e., $D^1_{\mathbf{d_1}}\Phi(v)$ lies in the tangent plane of Φ at $\Gamma(v)$ and it is not colinear with $\Gamma'(v)$. Analogously, we define a cross-boundary derivative $D^1_{\mathbf{d_2}}\varphi(v)$. Now, our condition for 'C^1' continuity is*

$$det\left[D^1_{\mathbf{d_1}}\Phi(v), D^1_{\mathbf{d_2}}\varphi(v), \Gamma'(v)\right] = 0. \tag{2.12}$$

We also call (2.12) a condition for the construction of 'visually continuous meshes'.

2.2 Punctual 'C^1'-continuity

We are going to obtain a theoretical result that guarantees the 'C^1' regularity at every vertex of a given mesh.

Proposition 3 *Two Bézier curves, γ_0, γ_1 defined over the interval $[0,1]$ by the control points $b^0_0, \cdots, b^0_n$ and $b^1_0, \cdots, b^1_n$ with $p = \gamma_0(1) = \gamma_1(0)$ are C^1 at the intersection point if and only if*

$$(b^0_n - b^0_{n-1}) = (b^1_1 - b^0_n).$$

Proof:

We can write γ_0 and γ_1 as:

$$\begin{aligned}\gamma_0(u) &= \sum_{i=0}^{n} b_i^0 B_i^n(u), \quad u \in [0,1],\\ \gamma_1(u) &= \sum_{i=0}^{n} b_i^1 B_i^n(u), \quad u \in [0,1],\end{aligned}$$

where $B_i^n(u)$ is the i-th Bernstein polynomial of degree n in one variable:

$$B_i^n(u) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i}, \quad u \in [0,1].$$

It is easy to verify that

$$B_i^{n\,\prime}(0) = \begin{cases} -n & \text{if } i = 0,\\ n & \text{if } i = 1,\\ 0 & \text{otherwise,}\end{cases}$$

$$B_i^{n\,\prime}(1) = \begin{cases} -n & \text{if } i = n-1,\\ n & \text{if } i = n,\\ 0 & \text{otherwise,}\end{cases}$$

so, we get

$$\gamma_0{}'(1) = \gamma_1{}'(0) \iff n(b_n^0 - b_{n-1}^0) = n(b_1^1 - b_0^1), \tag{2.13}$$

and from the continuity condition, $\gamma_0(1) = \gamma_1(0)$, we obtain that $b_n^0 = b_0^1$, which confirms the above result.

Remark 7 *It is important to note that colinearity of three distinct control points b_{n-1}^0, $b_n^0 = b_0^1$ and b_1^1* **is not sufficient** *to guarantee C^1-continuity. However, colinearity of these three points does implies that $\gamma_0{}'(1)$ is parallel to $\gamma_1{}'(0)$.*

Theorem 1 *If all control points around a vertex s_i, for every triangle K containing s_i as vertex, are coplanar, then the Bézier surface is 'C^1' at s_i and the plane of coplanarity shall be the common tangent plane at s_i, for all the surface patches containing s_i (see fig. (2.2)).*

Proof:

Let s_i be an arbitrary vertex of our mesh. If we fix in equation (2.5) a coordinate of the variable $\mathbf{u}$, u_k, we get a Bézier curve with one variable. This curve will be tangent to another

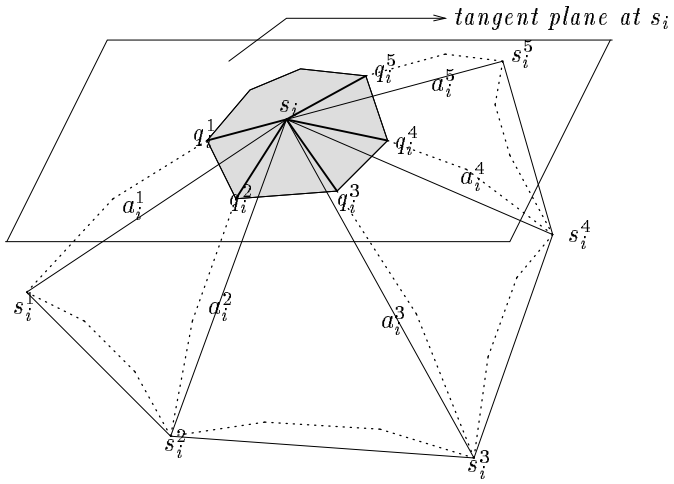

Figure 2.2: Control points around s_i

one contained in a neighboring patch if and only if the neighbor control points are colinear (cf. proposition (3)) with the common vertex.

Allcontrol points of these curves are linear combinations of the control points of the patches containing the vertex s_i and the coplanarity of the control points around s_i implies thus a continuous variation of the tangent vector for every curve. This fact guarantees the 'C^1'-continuity at s_i and the coplanarity plane of the control points around s_i is the tangent plane at s_i, for all the surface patches containing s_i.

2.3 Degree elevation

The two following proposition state that it is possible to express a Bézier curve or surface of degree n as one of order $n+1$. This process is named by *degree elevation* of a Bézier curve or surface.

Proposition 4 *Let $B_i^n(u)$ be the i-th Bernstein polynomial of degree n in one variable. If $\gamma(u)$ is defined by $\gamma(u) = \sum_0^n b_i B_i^n(u)$ then $\gamma(u) = \sum_0^{n+1} b_i^* B_i^{n+1}(u)$ with*

$$\begin{aligned} b_0^* &= b_0, \\ b_i^* &= b_{i-1}\frac{i}{n+1} + b_i\frac{n+1-i}{n+1}, \quad 1 \le i \le n, \\ b_{n+1}^* &= b_n. \end{aligned}$$

Proof: *We have:*

$$\begin{aligned}
\gamma(u) &= \sum_{i=0}^{n} b_i B_i^n(u) \\
&= \sum_{i=0}^{n} b_i \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} \\
&= \sum_{i=0}^{n} b_i \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i}(u+1-u) \\
&= \sum_{i=0}^{n} b_i \frac{n!}{i!(n-i)!} u^{i+1} (1-u)^{n-i} + \sum_{i=0}^{n} b_i \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i+1} \\
&= \sum_{i=0}^{n} b_i \frac{i+1}{n+1} \frac{(n+1)!}{(i+1)!(n-i)!} u^{i+1} (1-u)^{n-i} + \\
&\quad \sum_{i=0}^{n} b_i \frac{n+1-i}{n+1} \frac{(n+1)!}{i!(n+1-i)!} u^i (1-u)^{n-i+1} \\
&= \sum_{i=1}^{n+1} b_{i-1} \frac{i}{n+1} \frac{(n+1)!}{i!(n+1-i)!} u^i (1-u)^{n+1-i} + \\
&\quad \sum_{i=0}^{n} b_i \frac{n+1-i}{n+1} \frac{(n+1)!}{i!(n+1-i)!} u^i (1-u)^{n-i+1} \\
&= b_0 B_0^{n+1}(u) + \sum_{i=1}^{n} (b_{i-1} \frac{i}{n+1} + b_i \frac{n+1-i}{n+1}) B_i^{n+1}(u) + b_n B_{n+1}^{n+1}(u) \\
&= \sum_{i=0}^{n+1} b_i^* B_i^{n+1}(u).
\end{aligned}$$

Proposition 5 *Let $S(\mathbf{u})$ a triangular Bézier patch of order n with control points $\{b_I\}_{|I|=n}$. Then, it is possible to write $S(\mathbf{u})$ as a triangular Bézier patch of degree $n+1$:*

$$S(\mathbf{u}) = \sum_{|I|=n} b_I B_I^n(\mathbf{u}) = \sum_{|J|=n+1} b_J^* B_J^{n+1}(\mathbf{u}), \tag{2.14}$$

where

$$b_J^* = \frac{1}{n+1}[i_1 b_{J-e_1} + i_2 b_{J-e_2} + i_3 b_{J-e_3}], \quad |I| = n+1,$$

being $e_1 = (1,0,0)$, $e_2 = (0,1,0)$ and $e_3 = (0,0,1)$.

Proof:

$$
\begin{aligned}
S(\mathbf{u}) &= \sum_{|I|=n} b_I \frac{n!}{i_1!i_2!i_3!} u_1^{i_1} u_2^{i_2} u_3^{i_3} \\
&= \sum_{|I|=n} b_I \frac{n!}{i_1!i_2!i_3!} u_1^{i_1} u_2^{i_2} u_3^{i_3} (u_1+u_2+u_3) \\
&= \sum_{|I|=n} b_I \frac{n!}{i_1!i_2!i_3!} u_1^{i_1+1} u_2^{i_2} u_3^{i_3} + \sum_{|I|=n} b_I \frac{n!}{i_1!i_2!i_3!} u_1^{i_1} u_2^{i_2+1} u_3^{i_3} + \\
&\quad \sum_{|I|=n} b_I \frac{n!}{i_1!i_2!i_3!} u_1^{i_1} u_2^{i_2} u_3^{i_3+1} \\
&= \frac{1}{n+1} \left[\sum_{|I|=n} b_I (i_1+1) \frac{(n+1)!}{(i_1+1)!i_2!i_3!} u_1^{i_1+1} u_2^{i_2} u_3^{i_3} + \right. \\
&\quad \sum_{|I|=n} b_I (i_2+1) \frac{(n+1)!}{i_1!(i_2+1)!i_3!} u_1^{i_1} u_2^{i_2+1} u_3^{i_3} + \\
&\quad \left. \sum_{|I|=n} b_I (i_3+1) \frac{(n+1)!}{i_1!i_2!(i_3+1)!} u_1^{i_1} u_2^{i_2} u_3^{i_3+1} \right] \\
&= \sum_{|J|=n+1} \frac{1}{n+1} [i_1 b_{J-e_1} + i_2 b_{J-e2} + i_3 b_{J-e_3}] B_J^{n+1}(\mathbf{u}) \\
&= \sum_{|J|=n+1} b_J^* B_J^{n+1}(\mathbf{u}).
\end{aligned}
$$

2.4 Directional derivatives

Now, we are going to state some results related to directional derivatives of triangular Bézier patches and we also give a subdivision algorithm.

Let $\mathbf{d} = (d_1, d_2, d_3)$ be a vector. The directional derivative of a surface $S(\mathbf{u})$ with respect to the vector $\mathbf{d} = (d_1, d_2, d_3)$ is given by

$$
\begin{aligned}
D_{\mathbf{d}} S(\mathbf{u}) &= d_1 S_{u_1}(\mathbf{u}) + d_2 S_{u_2}(\mathbf{u}) + d_3 S_{u_3}(\mathbf{u}) \\
&= \sum_{|I|=1} \partial^I S(\mathbf{u}) B_I^1(\mathbf{d}),
\end{aligned}
$$

where

$$
\partial^I S(\mathbf{u}) = \frac{\partial^{|I|}}{\partial u_1^{i_1} \partial u_2^{i_2} \partial u_3^{i_3}} S(\mathbf{u})
$$

are the partials of S.
We can also compute higher order directional derivatives:

$$D^r_{\mathbf{d}} S(\mathbf{u}) = \sum_{|I|=r} \partial^I S(\mathbf{u}) B^r_I(\mathbf{d}). \tag{2.15}$$

We can also apply the operator $D^r_{\mathbf{d}}$ to Bernstein polynomials, i.e., we can combine (2.15) and (2.4):

$$D^r_{\mathbf{d}} B^n_I(\mathbf{u}) = \frac{n!}{(n-r)!} \sum_{|J|=r} B^r_J(\mathbf{d}) B^{n-r}_{I-J}(\mathbf{u}). \tag{2.16}$$

We are now in a position to give the r-th directional derivative of a triangular Bézier patch of order n.

Proposition 6 *Let S be a triangular Bézier surface patch of order n given by*

$$S(\mathbf{u}) = \sum_{|I|=n} b_I B^n_I(\mathbf{u}), \;\; u_1 + u_2 + u_3 = 1,$$

and let $\mathbf{d} = (d_1, d_2, d_3)$ be a vector. The r-th directional derivative of $S(\mathbf{u})$ with respect to $\mathbf{d}$ is given by

$$D^r_{\mathbf{d}} S(\mathbf{u}) = \frac{n!}{(n-r)!} \sum_{|J|=r} b^{n-r}_J(\mathbf{u}) B^r_J(\mathbf{d}). \tag{2.17}$$

Proof:

We apply (2.16) to the definition (2.10) and we obtain

$$\begin{aligned} D^r_{\mathbf{d}} S(\mathbf{u}) &= \frac{n!}{(n-r)!} \sum_{|I|=n} \sum_{|J|=r} b_I B^r_J(\mathbf{d}) B^{n-r}_{I-J}(\mathbf{u}) \\ &= \frac{n!}{(n-r)!} \sum_{|I|=n-r} \sum_{|J|=r} b_{I+J} B^r_J(\mathbf{d}) B^{n-r}_I(\mathbf{u}) \\ &= \frac{n!}{(n-r)!} \sum_{|J|=r} B^r_J(\mathbf{d}) \sum_{|I|=n-r} b_{I+J} B^{n-r}_I(\mathbf{u}). \end{aligned}$$

Now, the equation (2.9) completes the proof.

Remark 8 *A dual result is given by:*

$$D^r_{\mathbf{d}} S(\mathbf{u}) = \frac{n!}{(n-r)!} \sum_{|J|=n-r} b^r_J(\mathbf{d}) B^{n-r}_J(\mathbf{u}). \tag{2.18}$$

Remark 9 *We consider, now, the edge $u_1 = 0$ and a direction* $\mathbf{d}$ *not parallel to it. The directional derivative with respect to* $\mathbf{d}$*, evaluated along $u_1 = 0$ is* **the cross-boundary derivative**. *It is given by*

$$D^r_{\mathbf{d}} S(\mathbf{u})\Big|_{u_1=0} = \frac{n!}{(n-r)!} \sum_{|J_0|=n-r} b^r_{J_0}(\mathbf{d}) B^{n-r}_{J_0}(\mathbf{u})\Big|_{u_1=0}, \tag{2.19}$$

where $J_0 = (0, j_2, j_3)$.

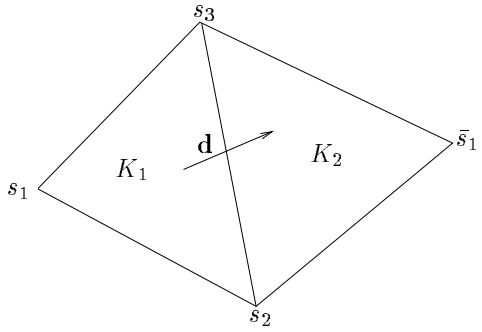

Figure 2.3: Subdivision: domain geometry.

2.5 Subdivision

Now, we consider a surface consisting of just two triangular Bézier surface patches of order n. Let their domain be defined by points s_1, s_2, s_3 and $\bar{s}_1$ (see fig. (2.3)), and their common boundary is through $s_2 s_3$. We suppose that a triangular Bézier surface patch S_b is given with the triangle $s_1 s_2 s_3$ as domain. It is also defined over $\bar{s}_1 s_3 s_2$. What are the Bézier points of S_b if we consider only the part of it that is defined over $\bar{s}_1 s_3 s_2$?

Proposition 7 (Subdivision algorithm). *Let K_1 and K_2 be two adjacent triangles with vertices s_1, s_2, s_3 and $\bar{s}_1$ (see fig. (2.3)). Let S_b be a triangular Bézier surface patch of degree n defined over K_1 , with control points $\{b_I\}_{|I|=n}$ .We suppose that the control points of S_b in the common edge $s_2 s_3$ are written by $I_0 = (0, i_2, i_3)$, $i_2 + i_3 = n$. Let $\bar{\mathbf{u}}$ denote the barycentric coordinates with respect to $\bar{s}_1 s_3 s_2$. Then $\bar{S}_b$, the extension of S_b over K_2, is a triangular Bézier surface patch of degree n and it can be described by*

$$\sum_{|I|=n} \bar{b}_I B^n_I(\bar{\mathbf{u}}), \tag{2.20}$$

where

$$\bar{b}_{I=(r,i_2,i_3)} = b^r_{(0,i_2,i_3)}(\mathbf{v}), \quad r = 0, \cdots, n \tag{2.21}$$

and $\mathbf{v}$ *is the barycentric coordinates of* $\bar{s}_1$ *with respect to* $s_1 s_2 s_3$.

Proof:

Let $\mathbf{u}$ denote the barycentric coordinates in $s_1 s_2 s_3$ and let $\bar{\mathbf{u}}$ denote those in $\bar{s}_1 s_3 s_2$. Then the surface $\bar{S}_b$ can be written in two ways:

$$\sum_{|I|=n} b_I B^n_I(\mathbf{u}) = \sum_{|I|=n} \bar{b}_I B^n_I(\bar{\mathbf{u}}).$$

Let $\mathbf{d}$ be the barycentric coordinates (with respect to $s_1 s_2 s_3$) of a direction that is not parallel to the common boundary $s_2 s_3$. Let $\bar{\mathbf{d}}$ be the barycentric coordinates of the same direction with respect to $\bar{s}_1 s_3 s_2$.
Now, we can consider directional derivatives with respect to $\mathbf{d}$ evaluated along the common edge ($u_1 = 0$) by applying (2.19) and we obtain:

$$\sum_{|I_0|=n-r} b^r_{I_0}(\mathbf{d}) B^{n-r}_{I_0}(\mathbf{u})\Big|_{u_1=0} = \sum_{|I_0|=n-r} \bar{b}^r_{I_0}(\bar{\mathbf{d}}) B^{n-r}_{I_0}(\bar{\mathbf{u}})\Big|_{\bar{u}_1=0}, \quad r = 0, \cdots, n,$$

where, $I_0 = (0, i_2, i_3)$. Comparing coefficients we have

$$b^r_{I_0}(\mathbf{d}) = \bar{b}^r_{I_0}(\bar{\mathbf{d}}), \quad r = 0, \cdots, n.$$

Here, $b^r_{I_0}$ and $\bar{b}^r_{I_0}$ are themselves polynomials of degree r. The last set of equations states that, for any r, the two polynomials agree in all derivatives up to order n, evaluated along the common boundary. So, they are equal for all values of $\mathbf{u}$ and $\bar{\mathbf{u}}$:

$$b^r_{I_0}(\mathbf{u}) = \bar{b}^r_{I_0}(\bar{\mathbf{u}}), \quad r = 0, \cdots, n.$$

Now, let $\mathbf{v}$ be the barycentric coordinates of $\bar{s}_1$ with respect to $s_1 s_2 s_3$. Noting that $\mathbf{v}$ correspond to $\bar{\mathbf{u}} = (1, 0, 0) = \bar{\mathbf{v}}$, we obtain

$$b^r_{I_0}(\mathbf{v}) = \bar{b}^r_{I_0}(\bar{\mathbf{v}}), \quad r = 0, \cdots, n.$$

Since $\bar{b}^r_{I_0}(\bar{\mathbf{v}}) = \bar{b}_{(r,i_2,i_3)}$, now we have

$$\bar{b}_{(r,i_2,i_3)} = b^r_{I_0}(\mathbf{v}), \quad r = 0, \cdots, n.$$

Remark 10 *This last proposition gives an algorithm that allows us to construct the Bézier points of the 'extension' of* S_b *to an adjacent patch.*

Remark 11 *If $\bar{s}_1$ is inside s_1, s_2, and s_3, then this algorithm uses convex combinations and (2.21) provides a subdivision algorithm.*

Remark 12 *Equation (2.21) gives a condition by which two adjacent patches of order n can be part of one global polynomial surface. If we do not let r vary from 0 to n, but from 0 to some $s \leq n$, we have a condition for C^s continuity between adjacent patches:*

$$\bar{b}_{I=(r,i_2,i_3)} = b^r_{(0,i_2,i_3)}(\mathbf{v}), \quad r = 0, \cdots, s.$$

This equation is a necessary and sufficient condition for the C^s continuity of two adjacent patches. We can make that claim since cross-boundary derivatives up to order s depend only on the $s+1$ rows of control points 'parallel' to the considered boundary. For $s = 1$ we obtain:

$$\bar{b}_{I=(1,i_2,i_3)} = v_1 b_{(1,i_2,i_3)} + v_2 b_{(0,i_2+1,i_3)} + v_3 b_{(0,i_2,i_3+1)}, \quad i_2 + i_3 = n - 1. \tag{2.22}$$

2.6 Global 'C^1'-continuity

Finally, we present the most important result that gives sufficient conditions to guarantee 'C^1'-continuity across a common boundary curve of two adjacent patches. Before stating it, we give some auxiliary lemmas that we use in its derivation.

Lemma 1 *Let*

$$\Gamma(u) = \sum_{i=0}^{n} b_i B_i^n(u), \quad u \in [0,1], \quad b_i \in \mathbb{R}^3,$$

then

$$\Gamma'(u) = n \sum_{i=0}^{n-1} (b_{i+1} - b_{i-1}) B_i^{n-1}(u).$$

Proof:

$$\begin{aligned}
\Gamma'(u) &= \sum_{i=0}^{n} b_i B_i^{n\,\prime}(u) \\
&= -nb_0(1-u)^{n-1} + \sum_{i=1}^{n-1} b_i \frac{n!}{i!(n-i)!}[iu^{i-1}(1-u)^{n-i} - (n-i)u^i(1-u)^{n-i-1}] + \\
&\quad b_n n u^{n-1}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} b_i n \frac{(n-1)!}{(i-1)!(n-i)!} u^{i-1}(1-u)^{n-i} - \sum_{i=0}^{n-1} b_i n \frac{(n-1)!}{i!(n-1-i)!} u^i (1-u)^{n-1-i} \\
&= \sum_{i=0}^{n-1} b_{i+1} n \frac{(n-1)!}{i!(n-1-i)!} u^i (1-u)^{n-1-i} - \sum_{i=0}^{n-1} b_i n \frac{(n-1)!}{i!(n-1-i)!} u^i (1-u)^{n-1-i} \\
&= n \sum_{i=0}^{n-1} (b_{i+1} - b_i) B_i^{n-1}(u).
\end{aligned}
$$

Lemma 2 *Let S_b be a triangular Bézier surface patch of order $n+1$ with control points b_I and let $\{L_u\}_{u\in[0,1]}$ be a family of curves defined by*

$$L_u(s) = \sum_{|I|=n+1} b_I B_I^{n+1}(\mathbf{g}(s)),$$

where $\mathbf{g}(s) = (s, (1-s)u, (1-s)(1-u))$, $s \in [0,1]$. Then

$$\frac{dL_u}{ds}(0) = (n+1)[\sum_{i=0}^{n} b_i^* B_i^n(u) - \sum_{i=0}^{n+1} \hat{b}_i B_i^{n+1}(u)],$$

where $b_i^* = b_{(1,i,n-i)}$ and $\hat{b}_i = b_{(0,i,n+1-i)}$.

Proof:

$$
\begin{aligned}
\frac{dL_u}{ds}(0) &= \sum_{\substack{I=(i_1,i_2,i_3)\\|I|=n+1}} b_I \frac{(n+1)!}{i_1!i_2!i_3!} \frac{d}{ds}[s^{i_1}(1-s)^{i_2}u^{i_2}(1-s)^{i_3}(1-u)^{i_3}](s=0) \\
&= \sum_{\substack{I=(i_1,i_2,i_3)\\|I|=n+1}} b_I \frac{(n+1)!}{i_1!i_2!i_3!}[i_1 s^{i_1-1}(1-s)^{n+1-i_1}u^{i_2}(1-u)^{i_3}](s=0) - \\
&\quad \sum_{\substack{I=(i_1,i_2,i_3)\\|I|=n+1}} b_I \frac{(n+1)!}{i_1!i_2!i_3!}[(n+1-i_1)s^{i_1}(1-s)^{n-i_1}u^{i_2}(1-u)^{i_3}](s=0) \\
&= \sum_{\substack{I=(i_1,i_2,i_3)\\|I|=n+1,\ i_1=1}} b_I \frac{(n+1)!}{i_2!i_3!} u^{i_2}(1-u)^{i_3} -
\end{aligned}
$$

$$
\begin{aligned}
& \sum_{\substack{I=(i_1,i_2,i_3)\\ |I|=n+1,\ i_1=0}} b_I \frac{(n+1)!}{i_2!i_3!}(n+1)u^{i_2}(1-u)^{i_3} \\
= & \sum_{i=0}^{n} b_{(1,i,n-i)}(n+1)\frac{n!}{i!(n-i)!}u^i(1-u)^{n-i} - \\
& \sum_{i=0}^{n+1} b_{(0,i,n+1-i)}(n+1)\frac{(n+1)!}{i!(n+1-i)!}u^i(1-u)^{n+1-i} \\
= & (n+1)[\sum_{i=0}^{n} b_i^* B_i^n(u) - \sum_{i=0}^{n+1} \hat{b}_i B_i^{n+1}(u)].
\end{aligned}
$$

Remark 13 *If the boundary $u_1 = 0$ is a Bézier curve of order n with control points $\{b_i\}_{0\le i\le n}$, i.e.,*

$$\sum_{i=0}^{n+1} \hat{b}_i B_i^{n+1}(u) = \sum_{i=0}^{n} b_i B_i^n(u)$$

then

$$\frac{dL_u}{ds}(0) = (n+1)\sum_{i=0}^{n}(b_i^* - b_i)B_i^n(u).$$

Lemma 3 *Let $b_i,\ b_i^*$ and $\bar{b}_i \in \mathbb{R}^3$, for $0 \le i \le n$, and*

$$
\begin{aligned}
\bar{b}_0 &= \alpha_1 b_0 + \alpha_2 b_1 + \alpha b_0^*, \\
\bar{b}_i &= \frac{n-i}{n}(\alpha_1 b_i + \alpha_2 b_{i+1} + \alpha b_i^*) + \frac{i}{n}(\alpha_3 b_{i-1} + \alpha_4 b_i + \alpha b_i^*), \quad 1 \le i \le n-1, \\
\bar{b}_n &= \alpha_3 b_{n-1} + \alpha_4 b_n + \alpha b_n^*,
\end{aligned}
$$

where $\alpha_1, \alpha_2, \alpha_3, \alpha \in \mathbb{R}$ are constants, $\alpha_1 + \alpha_2 + \alpha = 1$ and $\alpha_3 + \alpha_4 + \alpha = 1$. Let

$$
\begin{aligned}
\tau_0(u) &= n\sum_{i=0}^{n-1}(b_{i+1} - b_i)B_i^{n-1}(u), \quad u \in [0,1], \\
\tau_1(u) &= (n+1)\sum_{i=0}^{n}(b_i^* - b_i)B_i^n(u), \quad u \in [0,1], \\
\tau_2(u) &= (n+1)\sum_{i=0}^{n}(\bar{b}_i - b_i)B_i^n(u), \quad u \in [0,1].
\end{aligned}
$$

Then

$$\tau_2(u) = a_0(u)\tau_0(u) + a_1(u)\tau_1(u),$$

where

$$\begin{aligned} a_0(u) &= \frac{n+1}{n}[\alpha_2(1-u) - \alpha_3 u], \quad u \in [0,1], \\ a_1(u) &= \alpha, \quad u \in [0,1]. \end{aligned}$$

Proof:

$$\begin{aligned} a_0(u)\tau_0(u) &= (n+1)[\alpha_2(1-u) - \alpha_3 u] \sum_{i=0}^{n-1} (b_{i+1} - b_i) B_i^{n-1}(u) \\ &= -(n+1) \sum_{i=0}^{n-1} \alpha_3 (b_{i+1} - b_i) \frac{(n-1)!}{i!(n-1-i)!} u^{i+1}(1-u)^{n-i-1} + \\ &\quad (n+1) \sum_{i=0}^{n-1} \alpha_2 (b_{i+1} - b_i) \frac{(n-1)!}{i!(n-1-i)!} u^i (1-u)^{n-i} \\ &= -(n+1) \sum_{i=0}^{n-1} \alpha_3 (b_{i+1} - b_i) \frac{i+1}{n} \frac{n!}{(i+1)!(n-i-1)!} u^{i+1}(1-u)^{n-i-1} + \\ &\quad (n+1) \sum_{i=0}^{n-1} \alpha_2 (b_{i+1} - b_i) \frac{(n-i)}{n} \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} \\ &= -(n+1) \sum_{i=1}^{n} \alpha_3 (b_i - b_{i-1}) \frac{i}{n} \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} + \\ &\quad (n+1) \sum_{i=0}^{n-1} \alpha_2 (b_{i+1} - b_i) \frac{(n-i)}{n} \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} \\ &= (n+1) \left[\sum_{i=0}^{n-1} \alpha_2 (b_{i+1} - b_i) \frac{n-i}{n} B_i^n(u) - \sum_{i=1}^{n} \alpha_3 (b_i - b_{i-1}) \frac{i}{n} B_i^n(u) \right]. \end{aligned}$$

$$a_1(u)\tau_1(u) = (n+1) \sum_{i=0}^{n} \alpha (b_i^* - b_i) B_i^n(u).$$

If we note

$$T(u) = a_0(u)\tau_0(u) + a_1(u)\tau_1(u) + (n+1) \sum_{i=0}^{n} b_i B_i^n(u),$$

we obtain

$$\begin{aligned}
T(u) &= (n+1)\sum_{i=0}^{n-1}[\alpha_2(b_{i+1}-b_i)\frac{n-i}{n}+\alpha(b_i^*-b_i)+b_i]B_i^n(u)+ \\
&\quad -(n+1)\left[\sum_{i=1}^{n}\alpha_3(b_i-b_{i-1})\frac{i}{n}B_i^n(u)+(\alpha(b_n^*-b_n)+b_n)\,B_n^n(u)\right] \\
&= (n+1)\sum_{i=0}^{n-1}\frac{n-i}{n}[(1-\alpha_2-\alpha)b_i+\alpha_2 b_{i+1}+\alpha b_i^*]B_i^n(u)+ \\
&\quad (n+1)\sum_{i=1}^{n}\frac{i}{n}[(1-\alpha_3-\alpha)b_i+\alpha_3 b_{i-1}+\alpha b_i^*]B_i^n(u) \\
&= (n+1)\sum_{i=0}^{n-1}\frac{n-i}{n}[(\alpha_1 b_i+\alpha_2 b_{i+1}+\alpha b_i^*]B_i^n(u)+ \\
&\quad (n+1)\sum_{i=1}^{n}\frac{i}{n}[\alpha_3 b_{i-1}+\alpha_4 b_i+\alpha b_i^*]B_i^n(u) \\
&= (n+1)\sum_{i=0}^{n}\bar{b}_i B_i^n(u).
\end{aligned}$$

So, we get

$$a_0(u)\tau_0(u)+a_1(u)\tau_1(u)=(n+1)\sum_{i=0}^{n}(\bar{b}_i-b_i)B_i^n(u)=\tau_2(u).$$

Now we can obtain the following result that guarantees the 'C^1'-continuity across a common boundary curve of two adjacent patches.

Theorem 2 *(Farin)*
Let S_{b_1} and S_{b_2} be two adjacent patches of a continuous Bézier surface of order $n+1$. Let Γ be the common boundary curve (of order $n+1$) of S_{b_1} and S_{b_2} and suppose Γ can be considered as a Bézier curve of order n. Let $b_0,\ b_1,\cdots,\ b_n$ be the control points of Γ as a curve of order n and let $b_0^*,\ b_1^*,\cdots,\ b_n^*$ (respectively $\bar{b}_0,\ \bar{b}_1,\cdots,\ \bar{b}_n$) be the adjacent control points to Γ of S_{b_1} (respectively S_{b_2}) (see fig. (2.4)). Then a sufficient condition for 'C^1'-continuity of the Bézier surface across Γ is the following one:

There are $\alpha_1,\ \alpha_2,\ \alpha_3,\ \alpha_4,\ \alpha\in R$ verifying, $\alpha_1+\alpha_2+\alpha=1$ and $\alpha_3+\alpha_4+\alpha=1$, such that,

$$\bar{b}_0 = \alpha_1 b_0+\alpha_2 b_1+\alpha b_0^*, \tag{2.23}$$

$$\bar{b}_i = \frac{n-i}{n}(\alpha_1 b_i + \alpha_2 b_{i+1} + \alpha b_i^*) + \frac{i}{n}(\alpha_3 b_{i-1} + \alpha_4 b_i + \alpha b_i^*), \quad 0 < i < n, \quad (2.24)$$

$$\bar{b}_n = \alpha_3 b_{n-1} + \alpha_4 b_n + \alpha b_n^*. \quad (2.25)$$

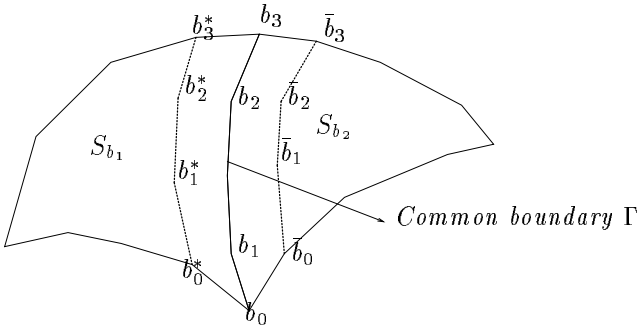

Figure 2.4: Control points for 'C^1'-continuity ($n = 3$).

Proof:

We can write Γ as a Bézier curve of order n as

$$\Gamma(u) = \sum_{i=0}^{n} b_i B_i^n(u), \quad u \in [0,1].$$

The tangent vector of Γ can be written by (see lemma (1))

$$\tau_0(u) = \Gamma'(u) = n \sum_{i=0}^{n-1} (b_{i+1} - b_i) B_i^{n-1}(u), \quad u \in [0,1].$$

Now, we suppose that Γ is the first boundary curve of S_{b_1} (i.e., obtained with $u_1 = 0$) and we consider the family of curves $\{\gamma_u^1\}_{u \in [0,1]}$ defined by:

$$\gamma_u^1(s) = \sum_{|I|=n+1} b_I^1 B_I^{n+1}(\mathbf{g}(s)),$$

where $\mathbf{g}(s) = (s, (1-s)u, (1-s)(1-u))$, $s \in [0,1]$ and b_I^1 are the control points of S_{b_1}. By construction, γ_u^1 is a curve in S_{b_1}, for every $u \in [0,1]$ and it verifies:

1. $\gamma_u^1(0)$ is in Γ, for all $u \in [0,1]$ (in fact, it is a parametrization of Γ in u),

2. $\gamma_u^1(1)$ is the opposite vertex to Γ in S_{b_1}.

So, the tangent vector at $\gamma_u^1(0)$ is a different tangent vector of S_{b_1} at every point of Γ and it can be expressed by (see lemma (2) and remark (13))

$$\tau_1(u) = \frac{d\gamma_u^1}{ds}(0) = (n+1)\sum_{i=0}^{n}(b_i^* - b_i)B_i^n(u).$$

(We note that $b_i^* = b^1_{(1,i,n-i)}, \quad 0 \le i \le n$).
If we use the same arguments with S_{b_2}, we have:

$$\tau_2(u) = (n+1)\sum_{i=0}^{n}(\bar{b}_i - b_i)B_i^n(u).$$

Now, the 'C^1'-continuity condition (2.12) takes the form

$$\det[\tau_0(u), \tau_1(u), \tau_2(u)] = 0. \tag{2.26}$$

We note that (2.26) is true if $b_{i+1} - bi$, $b_i^* - b_i$ and $\bar{b}_i - b_i$ are all coplanar. We exclude this trivial case and (2.26) is equivalent to the existence of coefficients $a_i(u)$ not all zero such that

$$a_0(u)\tau_0(u) + a_1(u)\tau_1(u) + a_2(u)\tau_2(u) = 0, \quad a_0(u),\ a_1(u),\ a_2(u) \neq 0. \tag{2.27}$$

In order to arrive at a manageable 'C^1' construction, we specify that $a_0,\ a_1,\ a_2$ be linear polynomials. It is clear that a_1 and a_2 must be constants, while a_0 must be linear. So, we obtain the result taking (see lemma (3)):

$$\begin{aligned} a_0(u) &= \frac{n+1}{n}(\alpha_2(1-u) - \alpha_3 u), \quad u \in [0,1], \\ a_1(u) &= \alpha, \quad u \in [0,1], \\ a_2(u) &= -1, \quad u \in [0,1]. \end{aligned}$$

Remark 14 *The equations (2.23) and (2.25) are equivalent to say that:*
$b_0,\ b_1,\ b_0^*$ *and* $\bar{b}_0$ *(respectively* $b_{n-1},\ b_n,\ b_n^*$ *and* $\bar{b}_n$*) are coplanar and*

$$\frac{|\bar{b}_0 b_1 b_0|}{|b_0^* b_0 b_1|} = \frac{|\bar{b}_n b_n b_{n-1}|}{|b_n^* b_{n-1} b_n|} \tag{2.28}$$

If we want to verify the equation (2.28), we consider, for example, equation (2.23)

$$\begin{aligned} \bar{b}_0 &= \alpha_1 b_0 + \alpha_2 b_1 + \alpha b_0^* \\ \bar{b}_0 - b_0 &= \alpha_1(b_0 - b_0) + \alpha_2(b_1 - b_0) + \alpha(b_0^* - b_0) \\ (\bar{b}_0 - b_0) \times (b_1 - b_0) &= \alpha(b_0^* - b_0) \times (b_1 - b_0). \end{aligned}$$

So, we obtain

$$\alpha = \frac{|\bar{b}_0 b_1 b_0|}{|b_0^* b_0 b_1|}.$$

Using the same argument for the equation (2.25), we have the above result.

3 Farin's algorithm

Below we outline Farin's algorithm for the construction of a 'visually continuous' Bézier surface over a given triangulation, where the tangent planes to the surface at the vertices are assumed to be known. Finally we prove that this algorithm verifies the conditions of Theorems (1) and (2).

3.1 Construction of triangular Bézier surface patches of order 3 over each triangle

Our purpose is to define, over each triangle of the mesh, cubic surface patches, so that, the resulting global surface shall be 'C^1' at every vertex of the triangulation. We remember that a cubic triangular surface patch is well defined with 10 control points (see fig. (3.1)). So, we shall construct these 10 control points as follows:

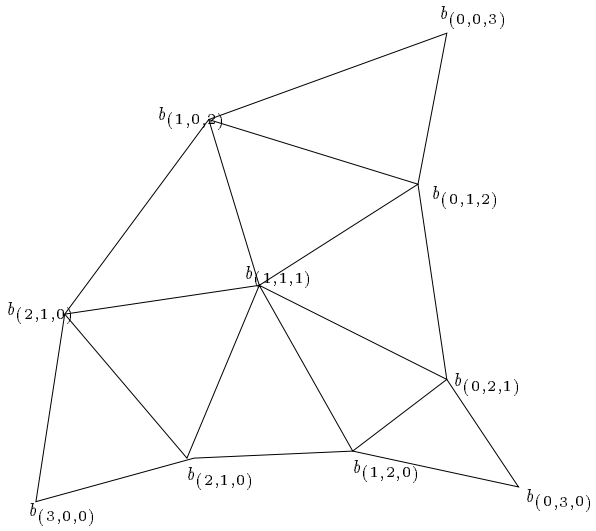

Figure 3.1: Control points b_I, $|I| = 3$, over K defining a surface patch of order 3.

1. Around each vertex, s_i, of the triangulation, we compute the control points $q_i^1, \cdots, q_i^{nv_{s_i}}$ (nv_{s_i} is the number of vertices around s_i) over the sides $a_i^1, \cdots, a_i^{na_{s_i}}$ (na_{s_i} is the number of mesh edges connected with s_i) (see fig. (2.2)), satisfying the following two conditions:

   - q_i^j, $j = 1, \cdots, nv_{s_i}$ lie on the tangent plane at s_i,

- $\frac{|s_i q_i^j q_i^{j+1}|}{|s_i s_i^j s_i^{j+1}|}$ = constant (independent of $j = 1, \cdots, nv_{s_i} - 1$ and s_i) which is taken as $\frac{1}{9}$. This condition will be used in order to guarantee equations (2.23) and (2.25) of Theorem (2).

2. From these control points around all the vertices, we obtain, for every triangle K control points b_I for $|I| = 3$ except $b_{(1,1,1)}$, which can be computed so that:

$$\begin{aligned} S_K^3(\mathbf{u_0}) &= \frac{1}{4}[b_{(2,1,0)} + b_{(1,2,0)} + b_{(0,2,1)} + b_{(0,1,2)} + b_{(1,0,2)} + b_{(2,0,1)}] - \\ &\quad \frac{1}{6}[b_{(3,0,0)} + b_{(0,3,0)} + b_{(0,0,3)}], \end{aligned} \tag{3.1}$$

where $S_K^3(\mathbf{u})$ is a triangular Bézier surface patch of order 3 over a triangle K of our mesh and $\mathbf{u_0} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

This formula gives quadratic precision if the control points around vertices were computed exactly.

The Bézier surface defined over the triangulation will be continuous, because adjacent triangles have same control points over their common boundary and 'C^1' at vertices, but it will not be necessarily 'C^1' across inter-element boundaries. In order to obtain 'C^1'-continuity we are going to use Theorem (2). We proceed in the following way:

3.2 Subdivision and degree elevation

1. We first subdivide the triangular Bézier surface patch over K, for all $K \in T_h$ (T_h the given mesh), at the barycenter of K into 3 subpatches of order 3 using the *subdivision algorithm* (see proposition (7)) and we compute 19 control points that define them (see fig. (3.2) and (3.1)) as follows:

$$\begin{array}{ll} b_0 = b_{(3,0,0)}, \quad b_1 = b_{(0,3,0)}, & b_2 = b_{(0,0,3)}, \quad b_9 = S_K^3(\mathbf{u_0}), \\ b_3 = b_{(2,1,0)}, \quad b_4 = b_{(1,2,0)}, & \text{similar for } b_5,\ b_6,\ b_7,\ b_8, \\ b_{10} = \frac{b_0+b_3+b_8}{3}, & \text{similar for } b_{12},\ b_{14}, \\ b_{16} = \frac{b_9+b_3+b_4}{3}, & \text{similar for } b_{17},\ b_{18}, \\ b_{11} = \frac{b_9+b_{16}+b_{18}}{3}, & \text{similar for } b_{13},\ b_{15} \end{array}$$

2. Then, each subpatch of degree 3 is expressed as a patch of order 4 using the degree elevation algorithm (see proposition (5)). We compute 31 control points, c_i, $0 \leq i \leq 30$, from the b_i's as follows (see fig (3.3)):

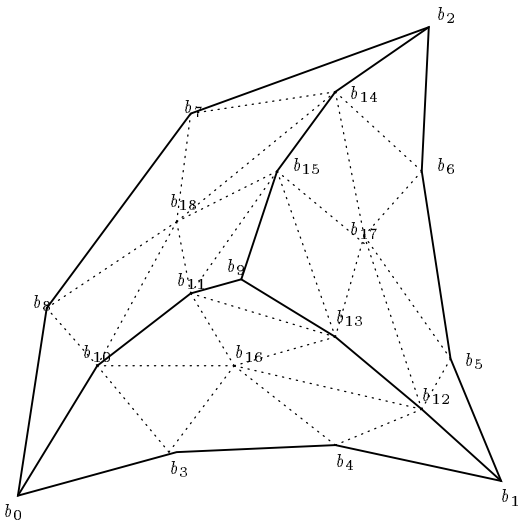

Figure 3.2: 19 control points defining the 3 subpatches of order 3.

$c_i = \quad b_i,\ i = 0, 1, 2; \qquad c_{12} = b_9;$

$c_3 = \quad (3b_3 + b_0)/4,\ c_4 = (2b_3 + 2b_4)/4,\ c_5 = (b_1 + 3b_4)/4,$ similar for $c_6,\ c_7,\ c_8,\ c_9,\ c_{10},\ c_{11};$

$c_{13} = \quad (3b_{10} + b_0)/4,\ c_{14} = (2b_{10} + 2b_{11})/4,\ c_{15} = (b_9 + 3b_{11})/4,$ similar for $c_{16},\ c_{17},\ c_{18},\ c_{19},\ c_{20},\ c_{21};$

$c_{22} = \quad (2b_{16} + b_{10} + b_3)/4,\ c_{23} = (2b_{16} + b_4 + b_{12})/4,\ c_{24} = (2b_{16} + b_{13} + b_{11})/4,$ similar for $c_{25},\ c_{26},\ c_{27},\ c_{28},\ c_{29},\ c_{30}.$

3.3 'C^1' corrections.

Now, for each inter-element boundary Γ of the original triangles, the interior control points $b_1^*,\ b_2^*$ and $\bar{b}_1,\ \bar{b}_2$ adjacent to Γ (see fig. (2.4)) are modified so that the condition (2.24) is satisfied. This modifies the control points numbered 22, 23, 25, 26, 28, 29 (see fig. (3.3)) for a triangle. For example, the control points numbered 22 and 23 are modified to obtain 'C^1'-continuity across the edge with control points 0, 3, 4, 5, 1. These modifications are computed in the following way:

Let S_{b_1} and S_{b_2} be two adjacent patches of order 4 and let Γ be their common boundary. Let $b_0,\ b_1,\ b_2$ and b_3 be the control points of Γ as a curve of order 3, that is before the degree

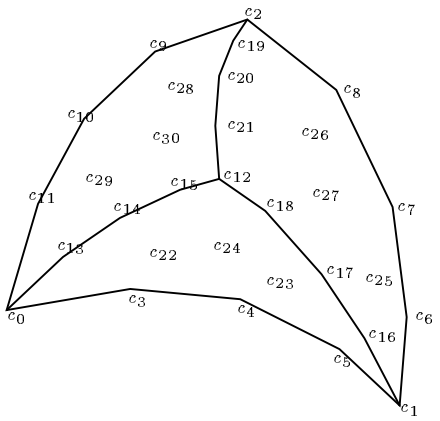

Figure 3.3: 31 control points defining 3 subpatches of order 4.

elevation action, and let $b_0^*,\ b_1^*,\ b_2^*$ and b_3^* (resp. $\bar{b}_0,\ \bar{b}_1,\ \bar{b}_2$, and $\bar{b}_3$) be the adjacent control points to Γ of S_{b_1} (resp. S_{b_2}) (see fig. (2.4)). We compute $\alpha_1,\ \alpha_2,\ \alpha_3,\ \alpha_4$ and α such that

$$\begin{aligned} \bar{b}_0 &= \alpha_1 b_0 + \alpha_2 b_1 + \alpha b_0^*, \quad \alpha_1 + \alpha_2 + \alpha = 1, \\ \bar{b}_3 &= \alpha_3 b_2 + \alpha_4 b_3 + \alpha b_3^*, \quad \alpha_3 + \alpha_4 + \alpha = 1. \end{aligned}$$

This is possible by the conditions of the subsection (3.1) and remark (14). Now, for $i = 1, 2$ let

$$\vec{\mu}_i = \frac{3-i}{3}(\alpha_1 b_i + \alpha_2 b_{i+1} + \alpha b_i^*) + \frac{i}{3}(\alpha_3 b_{i-1} + \alpha_4 b_i + \alpha b_i^*) - \bar{b}_i. \tag{3.2}$$

We compute $\vec{\phi}_i$ and $\vec{\chi}_i$ by

$$\vec{\phi}_i = \frac{1}{\alpha^2 + 1}\vec{\mu}_i, \tag{3.3}$$

$$\vec{\chi}_i = -\alpha\vec{\phi}_i, \tag{3.4}$$

and redefine b_i^* and $\bar{b}_i$ as $b_i^* + \vec{\chi}_i$ and $\bar{b}_i + \vec{\phi}_i$.
Then (2.24) holds for $i = 1, 2$ with $n = 3$. So we have 'C^1'-continuity of the Bézier surface across Γ.

The above modifications would disturb the interior 'C^1'-continuity of the original triangles. This is rectified by redefining the control points numbered 14, 17, 20, 15, 18, 21 and 12 in that order (see fig. (3.3)), for each K, as arithmetic averages of their surrounding 3 control points. For example,

$$c_{14} = (c_{13} + c_{22} + c_{29})/3, \quad c_{15} = (c_{14} + c_{24} + c_{30})/3. \tag{3.5}$$

3.4 Does this algorithm provide a 'C^1' Bézier surface?

Theorem 3 *The Bézier surface of order 4 over the mesh, obtained from the above algorithm is 'C^1'.*

Proof:

We are going to prove that the final control points verify the sufficient conditions stated in Theorems (1) and (2).

a) The Bézier surface is continuous by construction, that is, the boundary control points of every two adjacent patches are equal.

b) The control points of the cubic surface patches around a mesh vertex, s_i, are coplanar by construction (see subsection (3.1)) and the final control points of the triangular Bézier surface patches of order 4 around s_i are obtained (by the subdivision algorithm and the degree elevation process) as linear combinations of those of the cubic surface patches. So, they are coplanar, and Theorem (1) guarantees that our Bézier surface is 'C^1' at each vertex, s_i, of the triangulation.

c) Now we must check the 'C^1' regularity at each inter-element boundary of the mesh and also the interior 'C^1' continuity of the original triangles.

Let S_{b_1}, S_{b_2}, b_i, b_i^* and $\bar{b}_i$ $i = 0, \cdots, 3$ be as in the subsection (3.3) (see fig. (2.4)) , then, there are α_1, α_2, α_3, α_4 and $\alpha \in \mathbb{R}$ such that

$$\begin{aligned} \bar{b}_0 &= \alpha_1 b_0 + \alpha_2 b_1 + \alpha b_0^*, \quad \alpha_1 + \alpha_2 + \alpha = 1, \\ \bar{b}_3 &= \alpha_3 b_2 + \alpha_4 b_3 + \alpha b_3^*, \quad \alpha_3 + \alpha_4 + \alpha = 1. \end{aligned}$$

In effect, we obtain, thanks to the conditions of the subsection (3.1) and the subdivision and degree elevation algorithms, that the control points around a vertex s_i of our mesh are coplanar and

$$\frac{|\bar{b}_0 b_1 b_0|}{|b_0^* b_0 b_1|} = \frac{|\bar{b}_n b_n b_{n-1}|}{|b_n^* b_{n-1} b_n|},$$

and using remark (14), we check the conditions (2.23) and (2.25) of Theorem (2).

Noting that

$$\begin{aligned} b_i^* &= \hat{b}_i^* + \vec{\chi}_i, \quad i = 1, 2, \\ \bar{b}_i^* &= \hat{\bar{b}}_i + \vec{\phi}_i, \quad i = 1, 2, \end{aligned}$$

where

$$\begin{aligned}\vec{\phi}_i &= \frac{1}{\alpha^2+1}\vec{\mu}_i,\\ \vec{\chi}_i &= -\alpha\vec{\phi}_i,\end{aligned}$$

and

$$\vec{\mu}_i = \frac{3-i}{3}(\alpha_1 b_i + \alpha_2 b_{i+1} + \alpha\hat{b}_i^*) + \frac{i}{3}(\alpha_3 b_{i-1} + \alpha_4 b_i + \alpha\hat{b}_i^*) - \hat{\bar{b}}_i$$

condition (2.24) is obtained by a simple calculus and we have the 'C^1'-continuity of the Bézier surface across every inter-element boundary.

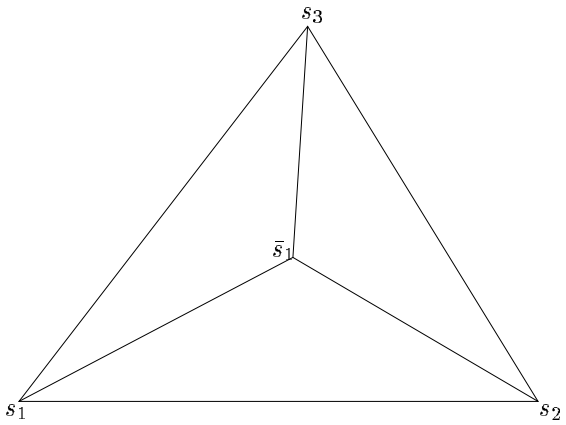

Figure 3.4: Original triangle $s_1 s_2 s_3$ and its three subpatches.

Now, we must check the 'C^1' regularity of the original triangles, i.e., the 'C^1'-continuity across a common boundary of two adjacent subpatches. But, the corrections that we have made using (3.5) correspond to equation (2.22) (see remark (12)) with $\mathbf{v} = (v_1, v_2, v_3) = (-1, -1, 3)$. We take this value of $\mathbf{v}$ because $\mathbf{v}$ is the barycentric coordinates of s_3 with respect to $s_1 s_2 \bar{s}_1$, where $\bar{s}_1$ is the barycenter of the original triangle $s_1 s_2 s_3$ (see fig. (3.4)) So, the Bézier surface of order 4 over our mesh is 'C^1'.

4 Implementation

4.1 Basic algorithm.

Here, we outline the computational procedure used to construct a 'C^1' Bézier surface for our problem starting from a given surface triangulation in $2D$ or $3D$ using Farin's algorithm. Here, the tangent planes at vertices are not given as input and they are computed approximately.

1. *Input:* We read as input the mesh, that is,
   - number of triangles, number of vertices,
   - connecting tables of triangles and vertices,
   - coordinates of the vertices,
   - references of vertices, edges and triangles.
2. *Data structure:* For the given triangulation we construct an internal data structure which gives:
   - for each vertex s_i, the vertices connected to it, and the triangles containing s_i,
   - the number of sides of the triangulation, and for each edge, a_i, the triangles containing it and the 2 vertices defining a_i.

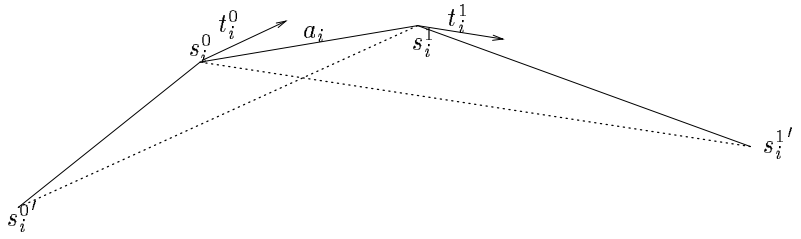

Figure 4.1: Construction of tangent vectors.

3. *Normal and tangent vectors:* If the normals to the surface at vertices of the mesh, T_h, are not given for each vertex s_i, a normal vector, $\vec{n}_i$, $1 \leq i \leq N_v$ is computed approximately as a weighted average of the unit outward vectors $\vec{n}_K$ of the triangles K containing s_i as follows:

$$\vec{n}_i = \frac{\sum_{K\,:\,s_i \in K} |K| \vec{n}_K}{\sum_{K\,:\,s_i \in K} |K|}. \tag{4.1}$$

We suppose, now, that our surface is not a closed one, that is, it is a surface with edges. For these edges, we must compute, for every side a_i, two tangent vectors, one for each vertex contained in a_i (see fig. (4.1)). We proceed as follows:

- t_i^0 is the tangent vector corresponding to s_i^0 and parallel to vector $s_i^{0\prime}s_i^1$ (see fig.(4.1)),
- t_i^1 is the tangent vector corresponding to s_i^1 and parallel to vector $s_i^0 s_i^{1\prime}$ (see fig.(4.1)).

4. *Construction of control points around vertices and control points for cubic patch over each triangle:* For any vertex s_i, let s_i^j, $j = 1, \cdots, nv_i$, be the vertices connected to s_i (see fig. (2.2)). We suppose that they are ordered from left to right with respect to s_i (see fig. (2.2)) and we assumed that s_i is an interior vertex, that is, $s_i^{nv_i} = s_i^1$. Let $\hat{s}_i^j$ be the projection of s_i^j in the tangent plane at s_i. If s_i is a boundary vertex and s_i^j is also a boundary vertex, we first project s_i^j over the tangent vector at s_i, corresponding to the edge $s_i s_i^j$, then, $\hat{s}_i^j$ is the projection of this intermediary point in the tangent plane at s_i. Let θ_i the angle between $s_i\hat{s}_i^j$ and $s_i\hat{s}_i^{j+1}$, $1 \leq j \leq nv_i - 1$. Now, we compute the lengths l_i, $1 \leq i \leq n$ as follows:

$$l_1 = \frac{1}{3}\text{length of } s_i s_i^1, \tag{4.2}$$

$$l_{j+1} = \frac{2}{9}\frac{|s_i s_i^j s_i^{j+1}|}{l_j \sin\theta_j}, \quad 1 \leq j \leq nv_i - 1. \tag{4.3}$$

Now, we compute q_i^j, $1 \leq j \leq nv_i$ (see fig. (2.2)), on $s_i\hat{s}_i^j$ with length l_j, $1 \leq j \leq nv_i$. If s_i is an interior point, i.e. when $s_i^1 = s_i^{nv_i}$, we want $q_i^1 = q_i^{nv_i}$. If it is not already satisfied, we modify q_i^1, $q_i^{nv_i}$ and $q_i^{nv_i-1}$ as follows:

Choose α, β such that for the points

$$\hat{q}_i^{nv_1-1} = q_i^{nv_i-1} + \alpha(q_i^{nv_i-2} - s_i), \quad \hat{q}_i^1 = q_i^1 + \beta(q_i^2 - s_i) \tag{4.4}$$

in order to obtain

$$\frac{|s_i\hat{q}_i^{nv_i-1}\hat{q}_i^1|}{|s_i s_i^{nv_i-1}s_i^1|} = \frac{1}{9}. \tag{4.5}$$

The equation (4.5) reduces to a quadratic equation in α if we put $\alpha = \beta$ or $\alpha = -\beta$ and can be solved. With the values of α and β thus obtained, we redefine $q_i^{nv_i-1}$, $q_i^{nv_i}$ and q_i^1 as $\hat{q}_i^{nv_i-1}$, $\hat{q}_i^1$ and $\hat{q}_i^1$ respectively.

The values of $|s_i q_i^{nv_i-2}q_i^{nv_i-1}|$ are not modified because $q_i^{nv_i-1}$ is moved parallel to $q_i^{nv_i-2} - s_i$, so, the control points q_i^j, $1 \leq j \leq nv_i$, satisfy the conditions noted at 1 of subsection (3.1). They are associated to the triangles connected to s_i as follows (see fig. (2.2)):

If K_i^j is the triangle $s_i s_i^j s_i^{j+1}$ with s_i as the k-th local vertex, then supposing $k = 1$, we take for K_i^j (see fig. (3.1))

$$b_{(3,0,0)} = s_i, \quad b_{(2,1,0)} = q_i^j, \quad b_{(2,0,1)} = q_i^{j+1};$$

the cases $k = 2, 3$ being similar.

Repeating this process for every vertex of the triangulation, we have, for each element K (see fig. (3.1)), all control points b_I, $|I| = 3$, except $b_{(1,1,1)}$, which is computed using (3.1).

5. *Subdivision:* Now, we subdivide the surface patch over each triangle K at the barycenter into 3 subpatches of degree 3 and we compute 19 local control points using the algorithm described in subsection (3.2).

6. *Degree elevation:* Then each subpatch of order 3 is expressed as a patch of order 4 and we compute 31 control points following the process given in subsection (3.2).

7. *'C^1' corrections:* Now, for every inter-element boundary Γ of the original triangles, we redefine the some interior control points to obtain 'C^1' regularity across each inter-element boundary. The process that we are followed is described in subsection (3.3).

8. *End:* The final surfaces is 'C^1' (see subsection (3.4)).

4.2 Surface with sharp edges.

For a surface with sharp edges, *we do not want 'C^1'-continuity across sharp sides.* If we impose that, the algorithm could give, some times, very distorted shapes near the sharp edges. This can be taken care of in the algorithm by the following modifications:

1. *Identification of sharp edges and fixed points.* Sharp edges can be identified as follows: For each inter-element boundary Γ, we compute the two unit outward normal vectors of the two triangles containing it. If the angle between them is greater than a value given by the user, then Γ is considered as a sharp edge. Now, we consider that a vertex is a fixed point of our mesh if:

   - it is the intersection of 3 or more sharp edges,
   - it is the intersection of 3 or more subdomains,
   - it is the intersection of 2 or more subdomains and it lies on the boundary,
   - the angle between two interconnected boundary edges, two interconnected sharp edges or two connected inter-subdomain edges is greater than the angle given by the user. In this case, the tangent vector, corresponding to the edges containing it at this vertex, is parallel to the respective edge,

- it is a cone vertex.

2. *Control points around vertices:* If s_i is a vertex on a sharp side, we proceed in the following way: The sharp edges divide the surface at s_i into different 'C^1' subpatches. We arrange the subpatches according to an angle criteria: first, the one with greater angular variation among the unit outwards normal vectors of the triangles in it. Now, for each subpatch, we compute, separately, the average normal vector of the triangles connected to s_i of that subpatch and then, the control points as in step 4, imposing over each sharp edge, the same control points in order to guarantee C^0 condition.

3. *No 'C^1' corrections:* The computation will be skipped for a sharp edge.

5 'Geometrically continuous' mesh generation

In this section, we present the code, developed at INRIA that generates 'geometrically continuous' meshes from a given triangulation, using the Farin's algorithm introuuced in the previous sections. The resulting mesh is finer than the original one, and it is obtained by subdividing each triangle in 4, 9, 16, ... new triangles. If we have a scalar or vector field over the original mesh, with values at the vertices, we also obtain a new field over the new vertices of the final mesh, using linear interpolation.

5.1 Data structure.

We outline, the different data required to construct a new mesh from a given one and a possible scalar or vector field solution over this new grid.

1. *Mesh dimension:* It is an integer, and the two admissible values are 2, if it is a 2D mesh, or 3 in the $3D$ case.

2. *Mesh type:* Specifies the type of our mesh. If the dimension is equal to 2 there is no choice, and the default one is **'.am_fmt'** used in other codes as EMC^2 [10] . In the other case, we can choose between two different mesh types: **'.am_fmt'** or **'.points and .faces'** . The $3D$ '.am_fmt' type is the natural generalization of the $2D$ type, and the '.points and .faces' type is the one implemented at MODULEF [1].

3. *Mesh name:* Here, we must give our mesh name, *without the type extension*, that is, if our mesh is named 'toto.am_fmt', we have to write 'toto'.

4. *Solution over the mesh? :* There are two possible values: 1 if we have defined a scalar or vector field over the mesh, 0 if not. If the answer of this question is affirmative, i.e. 1, then, we must give the solution name. This solution has to be given over the mesh vertices in **'.bb'** format, also used by *MODULEF* [1]. As in mesh name, we must specify the name, without extension. So, if our solution is named 'solution.bb' we have to write 'solution'.

5. *Final mesh name:* Here, we give the name of the final mesh. The mesh type is identical to the initial one, and as in that case, we don't specify the extension. So, if the initial mesh type is '.am_fmt ' and we write 'end_toto', we obtain 'end_toto.am_fmt'.

6. *Number of cuts for each mesh edge:* Here, we must type an integer value between 1 and 6 ($1 \leq nc \leq 6$). So, each initial triangle will be cut in nc^2 new triangles to obtain the final triangulation (see fig. (5.1)). We note, that if $nc = 1$ then the final and the initial triangulation are identical.

7. *Sharp edge angle:* Here, the user can introduce an angle expressed in degrees, θ, that controls the mesh sharp edges as follows: For each interior edge, Γ, we compute the

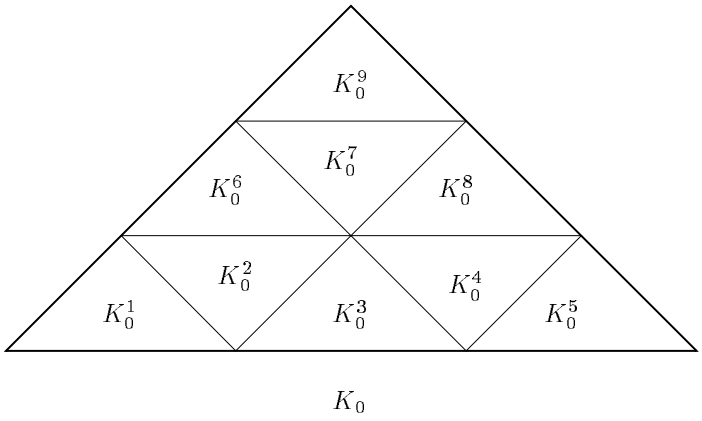

Figure 5.1: Initial triangle K_0 and the final ones with $nc = 3$.

two unit outward normal vectors of the two triangles on both sides of Γ, if the angle between them is greater than θ, Γ is considered as a sharp edge. We also use θ to determine fixed points, ex., if the angle between two connected boundary edges is greater than θ, then, their common vertex is a fixed point. If we introduce $\theta = 0$, then, angles are not considered.

This parameter is essential to keep the original geometry shape. If we suppose that we have a mesh of the unit square, and if we don't specify a suitable angle, i.e. $\theta < 90$ degrees, the final mesh will be rounded at the four corners (see fig. (5.2)).

5.2 General algorithm.

Below we outline the general algorithm to build 'geometrically continuous' meshes.

1. *Input data:* We introduce the data used by *ppinterpol* as listed in the previous subsection.

2. *Mesh input:* we create the internal data mesh structure, i.e. connection arrays, pointers between vertices, edges and triangles, etc, from the initial mesh. If we have also a solution over the mesh, we read it, and finally we check that the mesh does not contain bad or distorted elements (edges and triangles).

3. *Creation of the boundary curves:* Now, we examine each mesh edge, and if there is only one triangle containing it then it is a boundary edge. We construct a closed edge lists for each connected boundary components and we compute the associated tangent

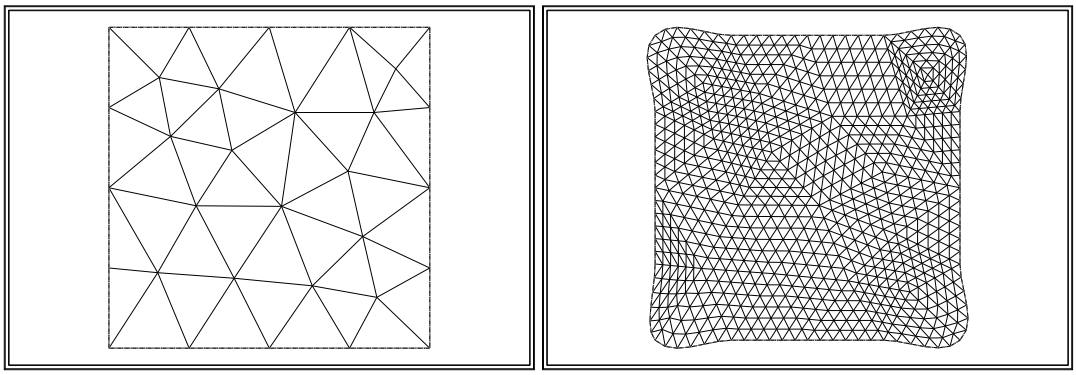

Figure 5.2: Initial mesh of the unit square and final distorted one, when we do not consider angles ($\theta = 0$).

vectors. It is important to note that **we can not work with meshes having edges contained in 3 or more triangles**[1]. We are now working to generalize the Farin's algorithm to this kind of meshes.

4. *Creation of inter-subdomain boundary curves:* This process is quite similar to the previous one. An edge Γ is named as an inter-subdomain edge if it is contained in two triangles with two different reference numbers. We construct edge lists for each inter-subdomain boundary and we also compute the associated tangent vectors.

5. *Creation of the Farin's algorithm control points:* Now, we use the Farin's algorithm presented in the previous sections to compute 31 control points over each triangle of the initial mesh.

6. *Creation of the final mesh:* At this step, we have computed the control points over each triangle, so we have a 'geometrically continuous' surface defined over the mesh. Now, we use the parameter nc given by the user to compute the number of points over each mesh triangle and the number of triangles obtained from the initial one. The number of points is equal to $(nc+1)(nc+2)/2$ and the number of triangles is equal to nc^2. Then, we identify the common points, that is, the points over each inter-element edges. Finally, for each point p over a triangle K_0 of the initial mesh, we compute the barycentric coordinates with respect to K_0. Farin's algorithm provides 31 control points over each triangle, corresponding to 3 subpatches of order 4. So, we have to determine which of these 3 subtriangles contains p and also its barycentric coordinates with respect to it. Now, knowing the subtriangle containing p, we choose the 15 control points defining a Bézier surface patch of order 4 (see fig. (3.3)). For

[1] This kind of meshes can appear in problems where we mix up $3D$ and $2D$ objects, but they can not be considered as surface meshes. They also appear at the intersection of different surfaces.

example, the control points corresponding to the first subpatch of order 4 are (see fig. (3.3)): c_0, c_3, c_4, c_5, c_1, c_{16}, c_{17}, c_{18}, c_{12}, c_{15}, c_{14}, c_{13}, c_{22}, c_{23} and c_{24}, similar for the others.

At this point, we use the equation (2.5) with $n = 4$ to compute the value of the surface at p, where $\mathbf{u}$ is the barycentric coordinates of p respect to the subtriangle and b_I the 15 control points.

The final result is a mesh with the two following properties:

- it is finer than the initial one if $nc > 1$,
- it is 'geometrically continuous'.

7. *Creation of the linear interpolation of the solution over the final mesh:* If we have a solution field associated to the initial mesh, we have to interpolate it over the final mesh. But, that is very simple, because, for each vertex of the final mesh, we know an initial triangle that contains it, and its barycentric coordinates.

8. *Output data:* Finally, we obtain the final mesh and the interpolated solution, if there exists, and they are written in the same data type that the initial ones.

5.3 Examples.

In this subsection we shows some examples that we have computed with *ppinterpol*.

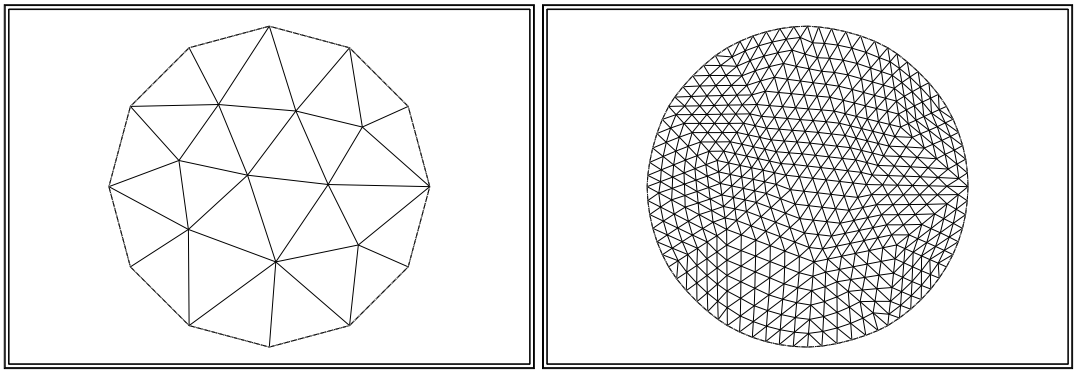

Figure 5.3: Initial mesh of the unit circle and final one.

1. *Unit circle:* The figure (5.3) shows the initial triangulation of the circle with 12 points in the boundary and the final result, with parameters $\theta = 0$ and $nc = 6$

2. *Plate with two domains:* This plate has two subdomains and the intersection line is a circle. We see with this example that we can obtain a better definition of the inter-subdomain boundary line. The figure (5.4) shows the initial mesh and the final one ($\theta = 60$ and $nc = 6$).

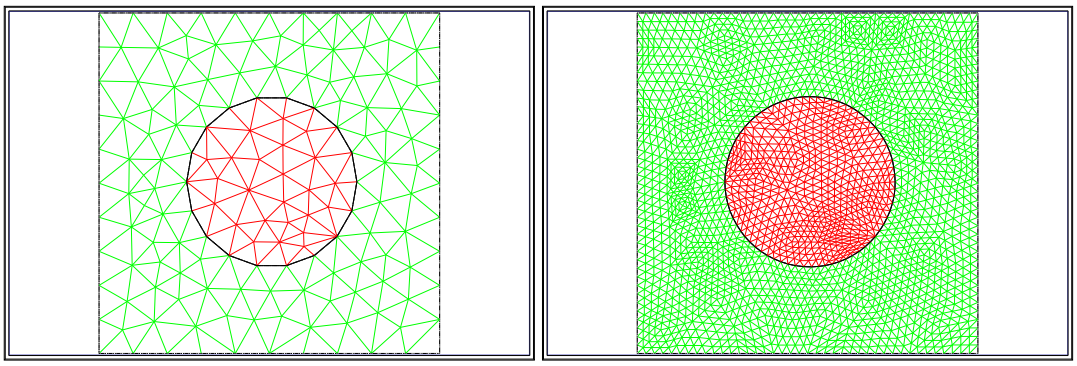

Figure 5.4: Initial mesh of a plate with two domains and final one.

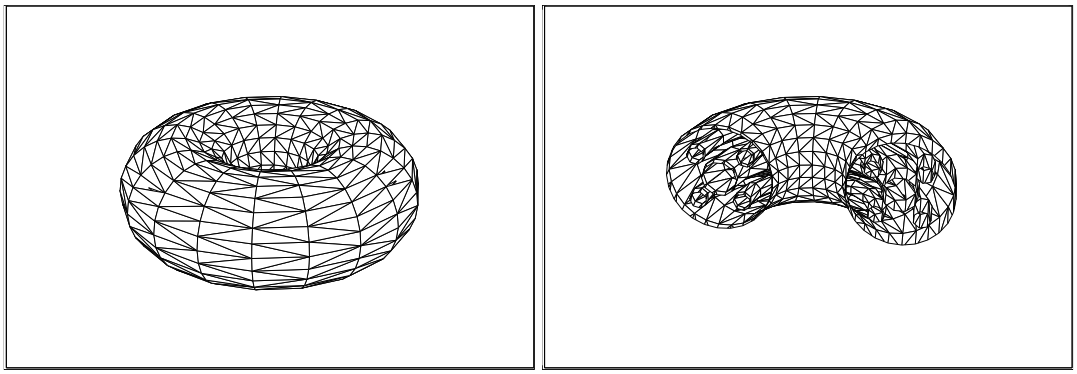

Figure 5.5: Initial mesh of a torus recovering 5 others.

3. *Six Torus:* In this case, the initial mesh is not connected. Figure (5.5) shows the initial triangulation (courtesy of E. Saltel,INRIA) and a cut of it, where we can see 6 different connected domains. Figure (5.6) shows the final result. Here $nc = 2$ and $\theta = 0$.

4. *Semi-sphere and semi-circle:* This example shows that we can work with surfaces made up $2D$ and $3D$ domains. Here we also have 6 different subdomains (see fig. (5.8)). The initial mesh and the final one are shown in figures (5.7) and (5.8). Here $\theta = 60$ and $nc = 4$.

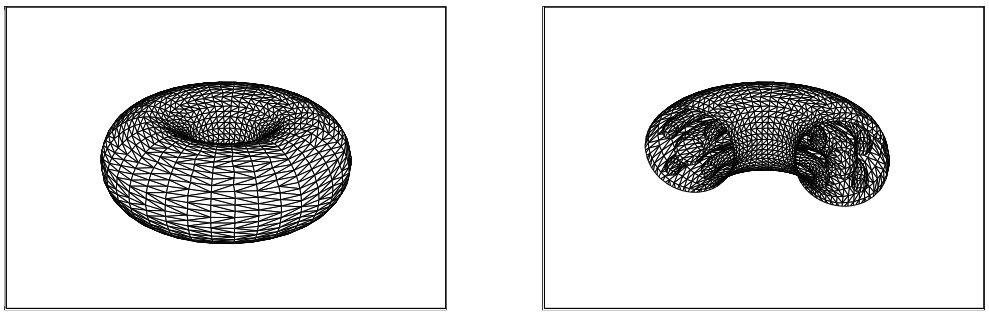

Figure 5.6: Final torus mesh with $nc = 2$ and $\theta = 0$.

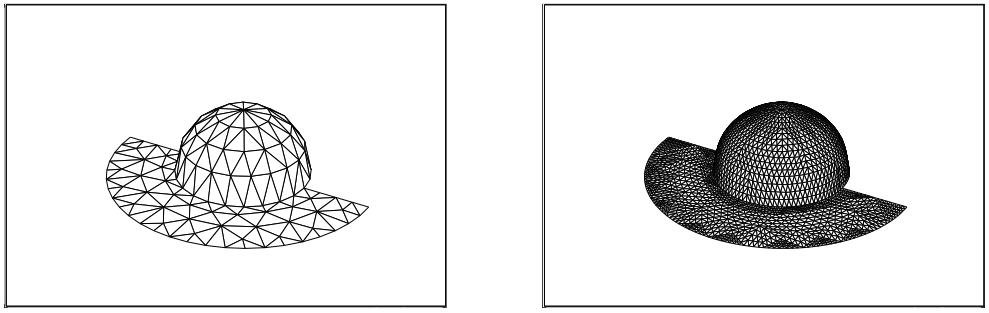

Figure 5.7: Initial and final mesh of a semi-sphere & semi-circle.

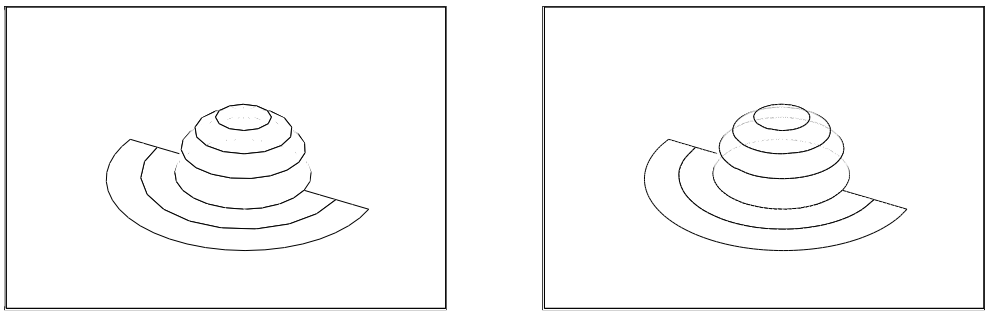

Figure 5.8: Inter-subdomain curves over initial and final mesh.

5. *Connecting rod:* Figure (5.9) shows the initial triangulation (courtesy E. Saltel, INRIA) of a connecting rod and the final mesh. Figure (5.10) shows an intersection line between two 'C^1' patches, and the final result. We can appreciate that the intersection line is also 'visually continuous'. Here $\theta = 30$ and $nc = 3$.

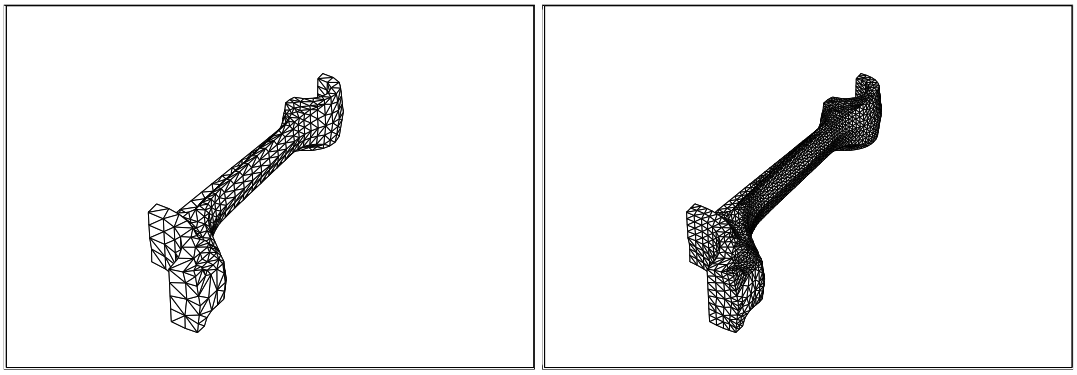

Figure 5.9: Initial mesh of a connecting rod and the final result obtained with $\theta = 30$ and $nc = 2$.

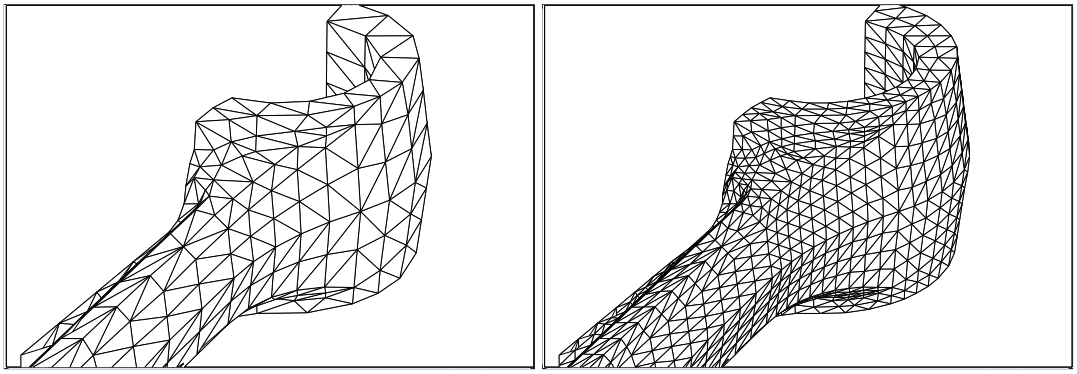

Figure 5.10: Zoom of the initial and final connecting rod.

6. *Airplane in a sphere:* Here, the mesh is not connected. Figure (5.11) shows a zoom of the initial triangulation (courtesy D.A) and the final mesh. In Figure (5.12), we can appreciate a detail of it. Here $\theta = 30$ and $nc = 2$.

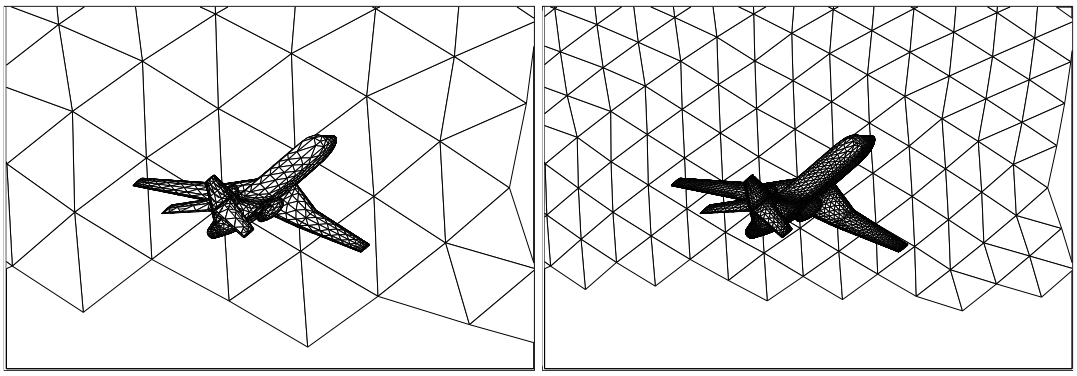

Figure 5.11: Zoom of the initial and final mesh of a sphere containing an airplane.

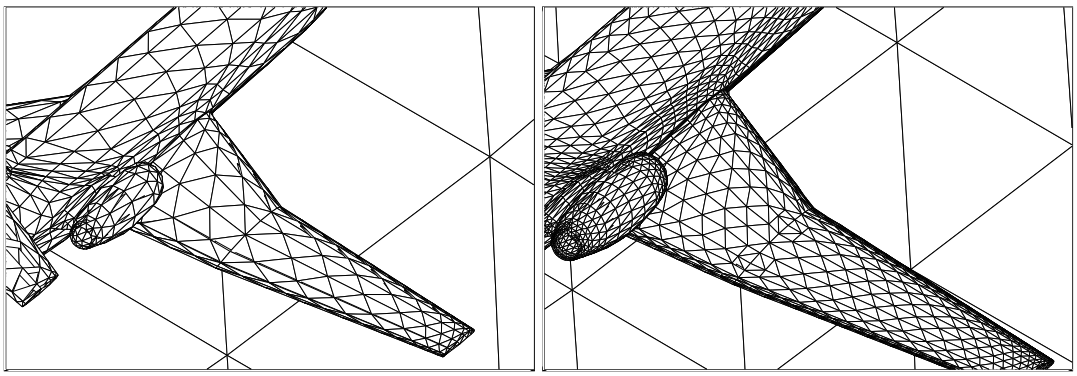

Figure 5.12: Detail of the initial and final airplane mesh.

References

[1] Bernadou et all., 'Modulef: une blibliothèque modulaire d'éléments finis'. INRIA (1988).

[2] W. Boehm, 'Generating the Bézier points of B-splines'. *Computer Aided Design*, **13**, no. 6: (1981).

[3] W. Boehm, G. Farin and J. Kahmann, 'A survey of curve and surface methods in CAGD'. *Computer Aided Geometric Design* **1**, no. 1: pp. 1–60, (1984).

[4] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*. North-Holland, 1977.

[5] G. Farin, 'A construction for the visual C^1 continuity of polynomial surface patches'. *Computer Graphics and Image Processing*, **20**: pp. 272–282, (1982).

[6] G. Farin, *Curves and surfaces for computer aided geometric design. A practical guide*. Academic Press, Inc., S. Diego, 1988.

[7] G. Farin, 'Smooth interpolation top scattered 3-D data'. *In R. Barnhill and W. Boehm, editors. Surfaces in Computer Aided Geometric Design*, North-Holland, (1982).

[8] G. Farin, 'Triangular Bernstein-Bézier patches'. *Computer Aided Geometric Design*. **3**, no. 2: pp. 83–128 (1986).

[9] S. Gopalsamy and O. Pironneau, 'Interpolation C^1 de resultats C^0'. *Rapport de recherche INRIA, Rocquencourt*, no. 1000 (Mars 1989).

[10] F. Hetch and E. Saltel, 'EMC^2 un logiciel d'édition de maillages et de contours bidimensionnels'. *Rapport Technique INRIA, Rocquencourt*, no. RT-0118 (Avril 1990).

[11] C. L. Lawson, 'Software for C^1 surface interpolation'. *Mathematical Software III*, J. R. Rice (ed.), Academic Press, New York, pp. 161–194, (1977).

[12] L. Piegl 'A CAGD theme: Geometric continuity of polynomial surface patches'. *Computer Aided Design*, **10**, no. 19: pp. 556–567, (1987).

[13] L. Piegl 'A geometric investigation of the rational Bézier scheme in computer aided geometric design'. *Computer in Industry*, **7**: pp. 401–410, (1987).

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399