A method for computing curved 2D and 3D meshes via the linear elasticity analogy: preliminary results

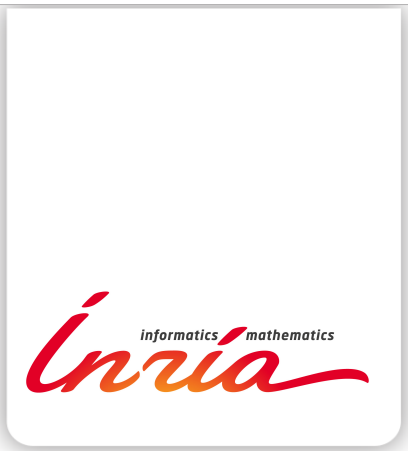
Remi Abgrall, Cécile Dobrzynski, Algiane Froehly

A method for computing curved 2D and 3D meshes via the linear elasticity analogy: preliminary results.

R. Abgrall, C. Dobzrynski , A. Froehly

RESEARCH REPORT
N° 8061
September 2012
Project-Team Bacchus

ISSN 0249-6399 ISRN INRIA/RR--8061--FR+ENG



A method for computing curved 2D and 3D meshes via the linear elasticity analogy: preliminary results.

R. Abgrall*, C. Dobzrynski †, A. Froehly ‡

Project-Team Bacchus

Research Report n° 8061 — September 2012 — 18 pages

Abstract: We propose, analyze an algorithm for the robust construction of curved meshes in two and three dimensions. The meshes are made of curved simplices. The algorithm starts from a mesh made of straight simplices, and using a linear elasticity analogy applied on well chosen data, one can generate a curved mesh. if the initial mesh has a boundary layer, the final mesh has also a boundary layer of equivalent quality.

Key-words: Calcul de maillages courbes, triangles et tétrahèdres courbes.

* Team Bacchus, INRIA, 200 chemin de la vieille tour, 33405 Talence cedex, France and IMB, Université de Bordeaux, 351 cours de la libération, 33405 Talence cedex France

† IMB, Université de Bordeaux, 351 cours de la libération, 33405 Talence cedex France and Team Bacchus, INRIA, 200 chemin de la vieille tour, 33405 Talence cedex, France

‡ Team Bacchus, INRIA, 200 chemin de la vieille tour, 33405 Talence cedex, France and IMB, Université de Bordeaux, 351 cours de la libération, 33405 Talence cedex France

RESEARCH CENTRE
BORDEAUX – SUD-OUEST

351, Cours de la Libération
Bâtiment A 29
33405 Talence Cedex

Une méthode de calcul de mailalges courbes bidimensionnels et tridimensionnels: rśultats préliminaire.

Résumé : Dans ce rapport, nous introduisons une méthode automatique permettant le calcul de maillages courbes valides partant d"une définition de la surface (courbe) et d'un maillage linéaire. Divers cas tests montrent que la méthode est robuste, et permet de gŕer les couches limites.

Mots-clés : Curved mesh generation, curve triangles, curved tetraherons.

1 Introduction

We are given a domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ which boundary can be parametrized by means of Bézier or Nurbs patches. In this work we assume that the patches are triangular (in 3D) or simple segments, see figure 1, in 2D. The aim of this paper is to propose and analyse an algorithm

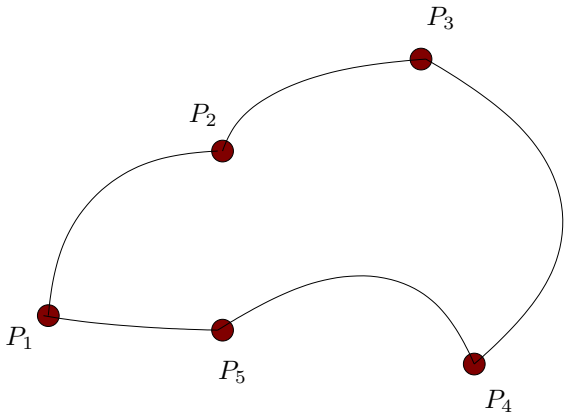


Figure 1: Example of a parametrized domain. For any $[P_i, P_{i+1}]$, the curve is parametrized by a Bézier or a Nurb curve.

that is able to mesh the domain Ω without any convexity assumption, while respecting *exactly* the original boundary.

There is a real interest in meshing computational domains with curved meshes. Indeed, there is a huge amount of work by many researchers throughout the world to design efficient and robust algorithms for fluid problems with accuracy formally higher than second order. If we specialize to the compressible fluid problems, examples are given by the Discontinuous Galerkin methods, see for example, among many other, [2, 3, 4], Finite element-like methods such as the SUPG method [9, 10], the isogeometric analysis method [11] or Residual distribution methods [1, 12]. In most of these methods (except the isogeometric analysis one), the parametrisation of the boundary is isoparametric. In [11], NURBS parametrisation are used, on quadrangular patches however.

It is known, for example since the work of Bassi et al., that the true accuracy of a high order method can only be revealed provided that the boundary are represented with at least the same accuracy.

If the construction of meshes by triangles and tetrahedra, while respecting the boundaries, is now a well mastered topic, see e.g. [7, 5], or GMSH [8], the construction of curved meshes is not a trivial matter, especially in the case of boundary layers and non convex domains.

The objective of this paper is to propose a robust algorithm that is able to construct such meshes. It relies on two ingredients: first a remark on Bézier and NURBS approximations, and then the appropriate use of a linear elasticity analogy applied on a well designed initial triangular/tet mesh. In this paper, we focus of the quadratic case, but the algorithm can be, *a priori*, extended to any order.

We first start by recalling a few classical informations on Bézier and NURBS approximation, then describe the algorithm, analyse it and then we provide examples in two and three dimensions.

2 Bézier and NURBS approximations.

In this paper, we have only considered Bézier and NURBS approximations with triangular patches. These approximations are defined as follows. We recall their structure on simplices, the case of linear, triangular and tetrahedral elements are only particular cases.

Let us consider $K \subset \mathbb{R}^d$ a simplex, i.e. the convex hull of $d+1$ independant points $\{P_1, \ldots, P_{d+1}\}$, i.e.

$$\text{Vol}(K) = \frac{1}{d+1} \left| \det(\overrightarrow{P_{d+1}P_1}, \ldots, \overrightarrow{P_{d+1}P_d}) \right| > 0. \tag{1}$$

We may assume that the numbering of the vertices is such that $\det(\overrightarrow{P_{d+1}P_1}, \ldots, \overrightarrow{P_{d+1}P_d}) > 0$. We can define barycentric coordinates, i.e for any $M \in \mathbb{R}^d$ a family of real numbers $\lambda_1(M), \ldots, \lambda_{d+1}(M)$ such that

$$\begin{aligned} \sum_{i=1}^{d+1} \lambda_i(M) &= 1 \\ M &= \sum_{i=1}^{d+1} \lambda_i(M) P_i. \end{aligned} \tag{2a}$$

These numbers are unique and given by the standard formula, where $j \neq i$ is arbitrary

$$\lambda_i(M) = \frac{\det\left(\overrightarrow{P_jP_1}, \ldots, \widehat{\overrightarrow{P_jP_i}}, \ldots, \overrightarrow{P_jP_d}\right)}{\det(\overrightarrow{P_jP_1}, \ldots, \overrightarrow{P_jP_d})} \tag{2b}$$

with, for any j,

$$\widehat{\overrightarrow{P_jP_i}} = \overrightarrow{P_jM}. \tag{2c}$$

Let us introduce a few notations. If $\alpha = (\alpha_1, \ldots, \alpha_{d+1})$ is a multi-index ($\alpha_i \in \mathbb{N}$), we define its lenght by $|\alpha| = \sum_{i=1}^{d+1} \alpha_j$. $\mathcal{I}$ is, in the sequel, the set of multi-indices of lenght n. If $\mathbf{x} = (x_1, \ldots, x_{d+1})$, we set

$$\mathbf{x}^\alpha := x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_{d+1}^{\alpha_{d+1}}.$$

We also define

$$\binom{|\alpha|}{\alpha} = \frac{|\alpha|!}{\Pi_{i=1}^{d+1} \alpha_i!}.$$

Let $c(\alpha)$ be the coefficient of x^α in the development

$$\Big(\sum_{i=1}^{d+1} x_i\Big)^n = \sum_{\alpha, |\alpha|=n} c(\alpha) \mathbf{x}^\alpha.$$

The Bézier polynomial is

$$B_\alpha^n(M) := c(\alpha) \Pi_{i=1}^{d+1} \lambda_i(M)^{\alpha_i}. \tag{3}$$

This family of polynomial defines a basis of $\mathbb{P}_n(x_1, \ldots, x_{d+1})$ which dimension is $N_d^n = \sum_{p=0}^{n} \binom{p+d-1}{p}$. It satisfies and satisfies

$$\begin{aligned} &B_\alpha^n(M) \geq 0 \qquad \text{for any } \alpha, |\alpha| = n \text{ and } M \in K \\ &\sum_{\alpha, |\alpha|=n} B_\alpha^n(M) = 1 \end{aligned} \tag{4}$$

Let us define the NURBS now. We consider a weight $\mathbf{w} = (\omega_{\alpha_1}, \ldots, \omega_{\alpha_{N_d^n}})$ with $\omega_\alpha > 0$ for all α, multi-index in $\mathbb{N}^{d+1}$ of lenght n. Then

$$N_\alpha^n = \frac{\omega_\alpha B_\alpha^n}{\sum\limits_{\alpha', |\alpha'|=n} \omega_{\alpha'} B_{\alpha'}^n}. \tag{5}$$

Again,

$$\begin{aligned} &N_\alpha^n(M) \geq 0 \qquad\qquad \text{for any } \alpha, |\alpha| = n \text{ and } M \in K \\ &\sum_{\alpha, |\alpha|=n} N_\alpha^n(M) = 1 \end{aligned} \tag{6}$$

In the following, we denote by φ_α^n either the functions B_α^n or N_α^n. They satisfy the relations (4) or (6),

$$\begin{aligned} &\varphi_\alpha^n(M) \geq 0 \qquad\qquad \text{for any } \alpha, |\alpha| = n \text{ and } M \in K \\ &\sum_{\alpha, |\alpha|=n} \varphi_\alpha^n(M) = 1 \end{aligned} \tag{7}$$

We are interested in the approximation of a function ψ defined on K with values in $\mathbb{R}^p$, $p > 0$. Given a family of control points $\mathcal{P} = \{P_\alpha \in \mathbb{R}^p, \alpha \in \mathcal{I}\}$, we approximate ψ by

$$\psi(M) \approx \sum_{\alpha, \alpha \in \mathcal{I}} P_\alpha \varphi_\alpha^n.$$

Of course the control parameters have to be chosen carefully for the approximation to be meaningful. We get immediately the following property

Property 2.1 (Convex hull property)**.** *For any $M \in K$, the relations* (7) *implies that $\psi(M)$ lies in the convex hull of $\mathcal{P}$.*

This property is not true in the case of Lagrange interpolation because the basis functions are not always positive, except in the case $n = 1$ where they are identical to the Bézier polynomials.

3 Descrition and analysis of the algorithm.

Let us further investigate the convex hull property 2.1 in the case $p = d$. In this case, a simplex K is mapped, via

$$M \in K \mapsto \psi(M)$$

onto a "curved" simplex $\hat{K}$, see Figure 2. On this figure, we have represented the points of barycentric coordinate $\alpha = (\frac{\alpha_1}{n}, \cdot, \frac{\alpha_d}{n})$ on K, and their image on $\hat{K}$. If the control points $\{P_j\}$ that are the image of the Lagrange points on the faces have disjoint convex hulls, then the images of the faces will not intersect, thanks to the property 2.1.

Considering for example the figure 3, where two simplices are represented, we see that under the same assumption the two elements images of mapping ψ_{K_1} and ψ_{K_2} will not intersect, since the image of a face is a face, and the mapping continuous. This remark is at the core of our method. We describe the idea on the simpler case of quadratic Bézier to begin with.

Consider a domain Ω with boundaries parametrized with Bézier functions. We first design a mesh with linear elements with one meshing tool. In our case, we have used `MMG3D` [6]. The idea is to deform this linear mesh. If one starts to deform the linear elements to produce curved elements, we may run into problems because curved faces may intersect. So instead of deforming

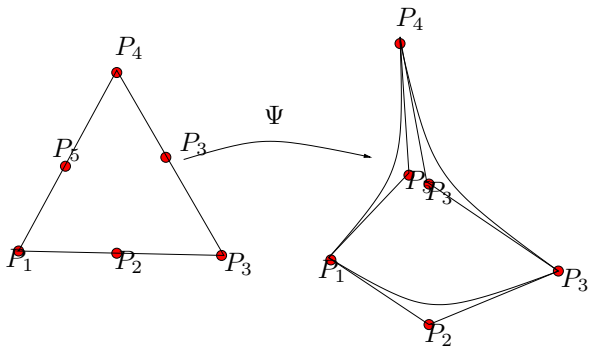


Figure 2: Transformation of K into $\hat{K}$ for a quadratic Bézier. The control points are represented.

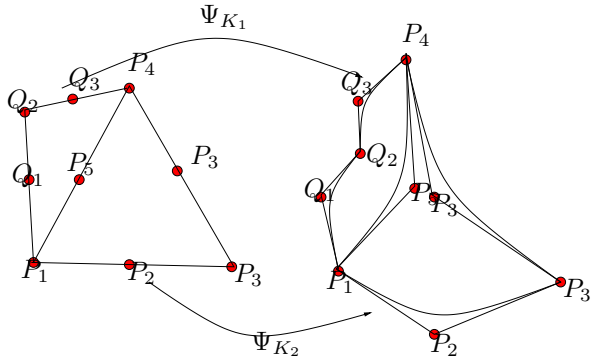


Figure 3: Transformation of two simplices for a quadratic Bézier. The control points are represented.

the linear element, we first subdivide them by adding the mid-points of edges, more generaly the Lagrange points of the elements. Here, we interpret them as control points. Then we subdivide the elements, see figure 4, and we deform the *subdivided* mesh. On the curved boundary, we impose that the control points that define the boundary are exactly recovered. If the new mesh is legal, with valid elements, then the mesh obtained from the new location of control point will also be legal, thanks to the property 2.1.

The main issue now is how to deform a linear mesh (obtained by subdivision of an initial $\mathbb{P}^1$ mesh), so that

- the boundary points of this mesh match to the control points defining the curved surface,
- All the elements of the deformed mesh are legal.

One way of achieving this goal can be to use a linear elasticity analogy. Consider a material with Lamé coefficients $\lambda > 0$ and $\mu > 0$. The solution can be obtained by solving on the initial mesh

$\mathcal{T}_h^0$ the linear elasticity problem

$$\begin{aligned} \text{div}\left(\lambda \text{tr}(\nabla u) + \mu(\nabla u + \nabla u^T)\right) &= 0 \text{ on } \Omega \\ u &= g \text{ on } \partial\Omega \end{aligned} \tag{8}$$

The deformed mesh $\mathcal{T}_h^D$ of control points is obtained by shifting the initial vertices of $\mathcal{T}_h^0$ of u, i.e.

$$M^D \text{ vertex of } \mathcal{T}_h^D \text{ if and only if there exists (a unique) } M^0 \in \mathcal{T}_h^0 \text{ such that } M^D = M^0 + u(M^0). \tag{9}$$

Thus the Dirichlet boundary condition g is obtained such that if $M^D \in \partial\Omega^D$, let $M^0 \in \partial\Omega^0$ the initial point before deformation

$$g(M^0) := M^D - M^0.$$

In general, the deformed mesh is not legal because the deformation is too large. To overcome this problem, we notice that the problem (8) is linear. Let us denote the mapping between g and u by $u = \mathcal{L}(g)$. If $\theta \in \mathbb{R}$, we have

$$\theta u = \mathcal{L}(\theta g).$$

A mesh is legal if, for any element $K^0 = convex(A^0_{i_1}, \ldots, A^0_{i_{d+1}})$ of $\mathcal{T}_h^0$ the set

$$K^D = convex(A^0_{i_1} + u(A^0_{i_1}), \ldots, A^0_{i_{d+1}} + u(A^0_{i_{d+1}}))$$

has the same orientation as the K^0. This condition is violated in particular when the volume of K^D changes sign. Considering the mapping

$$\omega_{K_0} : \theta \mapsto \text{Vol}\Big(convex(A^0_{i_1} + \theta u(A^0_{i_1}), \ldots, A^0_{i_{d+1}} + \theta u(A^0_{i_{d+1}}))\Big),$$

we see that $\omega_{K^0}(0) = vol(K^0) > 0$, and this it is enough to find the smallest θ for which there exists one K^0 for which $\omega_{K_0}(\theta) = 0$. This amounts to solving a quadratic (in 2D) or a cubic (in 3D) polynomial on all the simplices of $\mathcal{T}^0$ and look for the smallest root.

For obvious numerical reasons, we cannot accept elements with zero volume: it is necessary to strenghen a bit the previous criteria: if ω_{min} is the smallest volume of elements on the initial mesh, we impose $\omega_{K^0}(\theta) > \alpha\, \omega_{min}$. In practise, we choose $\alpha = 0.9$.

The algorithm is thus the following:

1. Solve for g,
2. Look for the smallest θ, say θ_0 such that for any $\theta \in [0, \theta_0[$, $\omega_{K^0}(\theta) > \alpha\, \omega_{min}$ for any K^0 in $\mathcal{T}^0$, then
   - if $\theta_0 > 1$, the final mesh is obtained,
   - if $\theta_0 < 1$, then update the mesh following (9), denote this new mesh as T^0 and go to step 1.

We cannot prove if the algorithm is converging, but all the experiments we have done show a very fast convergence, even for very deformed final meshes. The next section reports some of the tests we have done.

4 Examples.

In this work, we use S_J, the scaled Jacobian measure defined in [5] to evaluate the distortion of a curved element.

We note ξ the coordinates in the parametric space, E an element of this space and $T(\xi)$ the transfomation between E and the element (which may be curved or not) in the physical space. We denote by J_T the Jacobian of T and $|J_T|$ the determinant of this Jacobian. Then, the scaled Jacobian measure is defined by:

$$S_J = \frac{\min\limits_{\xi \in E} |J_T(\xi)|}{\max\limits_{\xi \in E} |J_T(\xi)|} \in [0, 1] \tag{10}$$

Since the Jacobian of the transformation is constant for a straight element, $S_J = 1$ in this case. The closer S_J from 1, the less distorted the element is. If S_J is close to 0, this means that some element is very slim. If $S_J < 0$, the element is not valid (negative volume).

4.1 Curved mesh examples.

The most of isotropic $\mathbb{P}^1$ mesh can certainly be transformed into a curved mesh without any problem if the deformation between the piecewise linear boundary and the curved one is not too large because the curvature of the boundary does not create non-valid elements. However, using linear elasticity analogy allows to generate better mesh, and as we see in this section, the method is quite robust.

4.1.1 First isotropic mesh.

We first consider a mesh represented on the Figure 5(a). The domain is non convex. This linear mesh contains 1616 vertices and 790 triangles. When we curved *only* the boundary, we obtain a mesh with 10 invalid triangles. An example of invalid triangle is represented on Figure 5(c). We apply the algorithm described in section 3 to generate a valid curved mesh. We need 9 iterations to move the boundary to its final position and we obtain the mesh shown on figure 5(b).

The histograms of the scaled Jacobian measure S_J are shown in figure **??**. On the histogram figure 6(a) (corresponding to the mesh of figure 5(c)), we see that the invalid mesh obtained by deforming the boundary only has several invalid curved elements. The quality of many of the other elements is not good because the scaled Jacobian is very close to 0 or negative for several elements. Using our algorithm, see figure 6(b), we do not have any invalid elements and the quality of the curved elements is better: the value of the scaled Jacobian is far from 0, and we also see that many elements are curved since the S_J is smaller than 1.

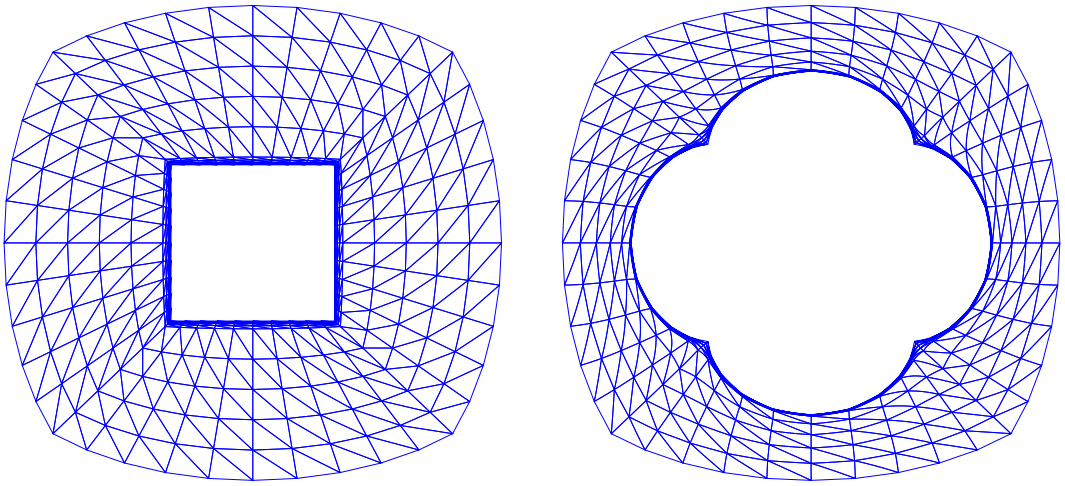
4.2 Boundary layers.

Let us consider now examples of mesh with boundary layers. This is *a priori* more challenging if we want to preserve the structure of the boundary layer mesh. The original mesh consists, near the boundary, of succession of layered cells. We want to keep this structure.

In a first example, the initial mesh is represented on figure 7(a). It has 400 vertices, 720 triangles. We want to map the square onto a treffle like shape, see figure 7(b). The final mesh is represented on figure 7(b). The deformation is so large that one cannot only deform the boundary.

The mesh represented on figure 7(b) is computed with our algorithm. We need 6 iterations only. All the elements are valid and most of them are straight i.e. the scaled Jacobians are larger that 0.9: 704 elements over the 720 elements of the mesh, see figure 6(b).

The figure 8 shows a zoom on a part of the Figure 7(b). We see that the structure of the initial mesh is conserved.



(a) Initial straight mesh. (b) Final curved mesh.

Figure 7: Boundary layers test case.

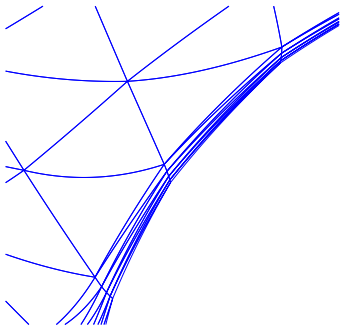


Figure 8: Zoom on boundary layer test case.

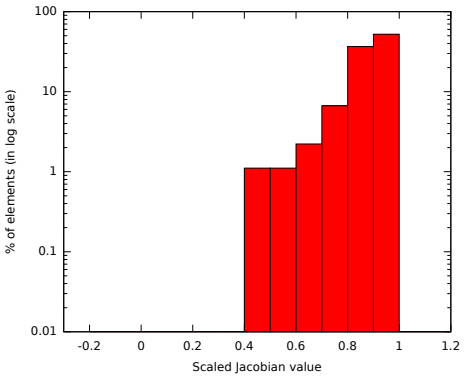


Figure 9: Boundary layers test case: Scaled Jacobians of last curved mesh.

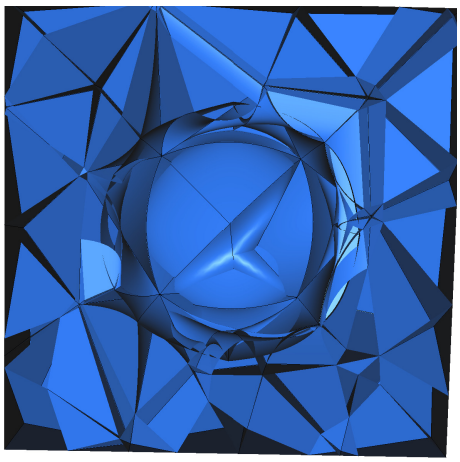
4.3 RAE 2822 airfoil.

The next example is the geometry of a RAE 2822 airfoil. We start from a fine structured mesh which is already well adapted to the geometry, but the aspect ratio of the element can be large (espcially in the flap parts of the airforil). With one iteration of the algorithm, we obtain the final mesh represented on Figure 10(a). Most of elements are good one (scaled jacobians ¿ 0.9) .

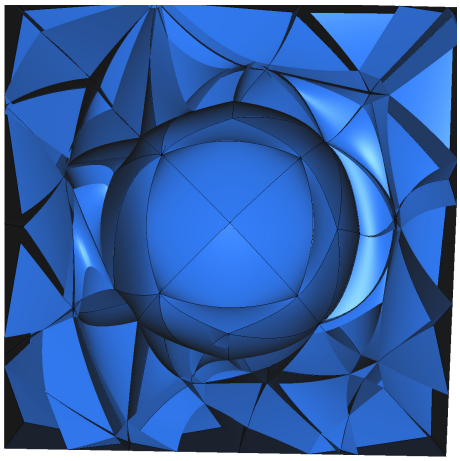
4.4 3D examples.

We give the 3D analog of the example 4.2. We start from a mesh where the internal boundary is the cube $[0,3]^3$. Each of its faces is deformed into the portion of a sphere of radius $\frac{\sqrt{3}}{16}$.

The internal curved surface crosses the elements of the initial mesh **??**: to show this, figure 11(a) shows a planar cut of the initial mesh when the curved boundary is also represented: wee see that intersection exist. On figure 11(b), we have represented the same planar section on the final mesh. There is no more intersections and we can see that the internal tetrahedra are curved.



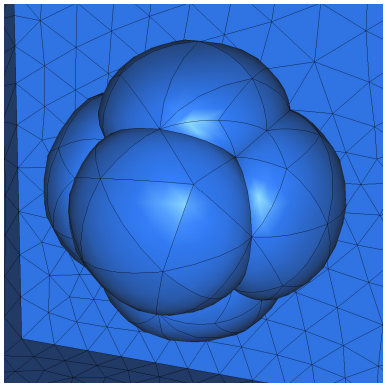
(a) initial tet mesh with the curved boundary



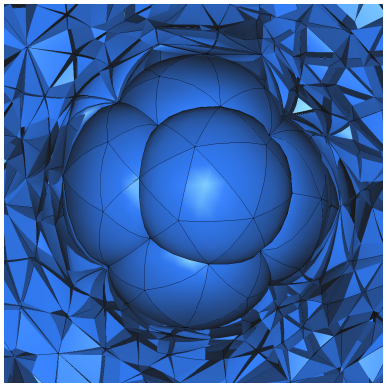
(b) final curved tet mesh

Figure 11: Planar cross section of the initail and final meshes.

On figure 12, the same test case is performed but the radii of the sphere are diferent from one face to another.



(a) Surface



(b) Planar cut

Figure 12: Same case as on figure 11 with different radii.

5 Conclusions and perspectives.

In this preliminary work we have developped an algorithm that is able to deform linear $\mathbb{P}^1$ meshes. It assumes that the exact boundary surfaces are known. We have performed several cases in 2 and 3 dimension. The quality of the elements is good, including when boundary layers exist. We are also able to preserve the structure of boundary layers. Further tests are needed to assess the robustness of the approach.

A High order mesh file

To manage high order meshes, we use the ASCII `MESH` format with some added keywords:

- `Order k` with k the order of the mesh.
- `NumberOfP1Dofs ndof` with ndof the number of degree of freedom of the associated P^1 mesh.
- `Weights`
  `np`

w_1
...
w_{np}

- `PkEdges`
- `PkTriangles`

Acknowledgements.

R. Abgrall has been partialy supported by the FP7 Advanced Grant # 226316 "ADDECCO".
A. Froehly has been fully supported by the FP7 Advanced Grant # 226316 "ADDECCO".

References

[1] R. Abgrall, A. Larat, and M. Ricchiuto. Construction of very high order residual distribution schemes for steady inviscid flow problems on hybrid unstructured meshes. *J. Comput. Phys.*, 230(11):4103–4136, 2011.

[2] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2D Euler equations. *J. Comput. Phys.*, 138(2):251–285, 1997.

[3] Bernardo Cockburn and Chi-Wang Shu. The Runge-Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws. *RAIRO, Modélisation Math. Anal. Numér.*, 25(3):337–361, 1991.

[4] Bernardo (ed.) Cockburn, George E. (ed.) Karniadakis, and Chi-Wang (ed.) Shu. *Discontinuous Galerkin methods. Theory, computation and applications. 1st international symposium on DGM, Newport, RI, USA, May 24–26, 1999.* Berlin: Springer, 2000.

[5] S. Dey, R.M. O'Bara, and M.S. Shephard. Towards curvilinear meshing in 3d: the case of quadratic simplices. *Computer-Aided Design*, 33(3):199 – 209, 2001.

[6] C. Dobrzynski and P. Frey. Mmg3d: Anisotropic tetrahedral remesher/moving mesh generation. http://www.math.u-bordeaux1.fr/ cdobrzyn/logiciels/mmg3d.php.

[7] P.L. George and H. Borouchaki. Construction of tetrahedral meshes of degree two. *Int. J. Numer. Methods Eng.*, 90(9):1156–1182, 2012.

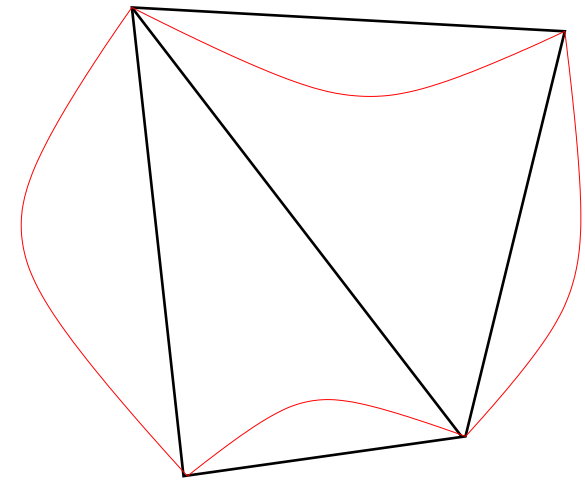
[8] Christophe Geuzaine and Jean-Franois Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. `http://geuz.org/gmsh/`.

[9] Thomas J.R. Hughes and Michel Mallet. A new finite element formulation for computational fluid dynamics. III: The generalized streamline operator for multidimensional advective-diffusive systems. *Comput. Methods Appl. Mech. Eng.*, 58:305–328, 1986.

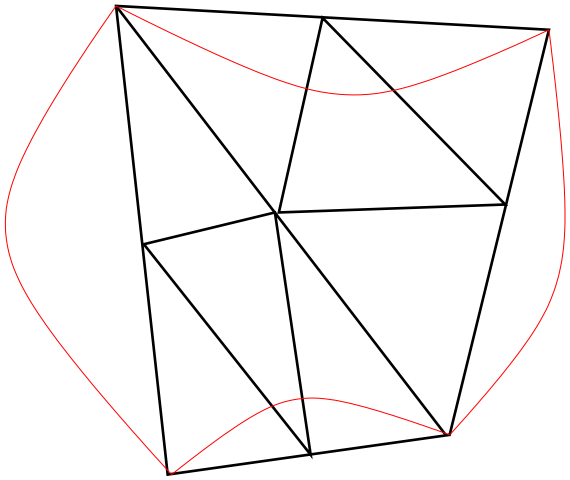
[10] Thomas J.R. Hughes, Michel Mallet, and Akira Mizukami. A new finite element formulation for computational fluid dynamics. II. Beyond SUPG. *Comput. Methods Appl. Mech. Eng.*, 54:341–355, 1986.

[11] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comput. Methods Appl. Mech. Eng.*, 194(39-41):4135–4195, 2005.

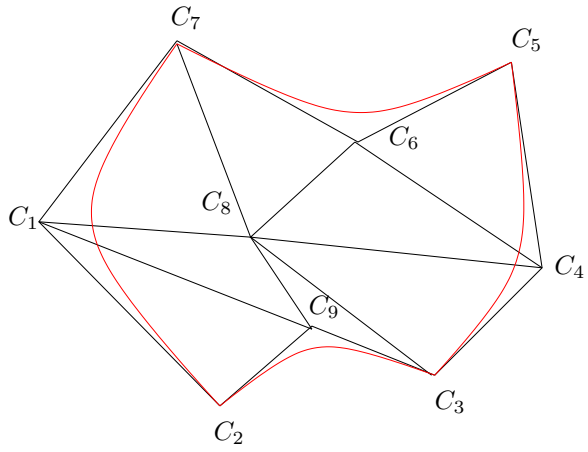
[12] Martin Vymazal, Tiago Quintino, Nadège Villedieu, and Herman Deconinck. High-order upwind residual distribution schemes on isoparametric curved elements. *J. Comput. Phys.*, 230(4):890–906, 2011.



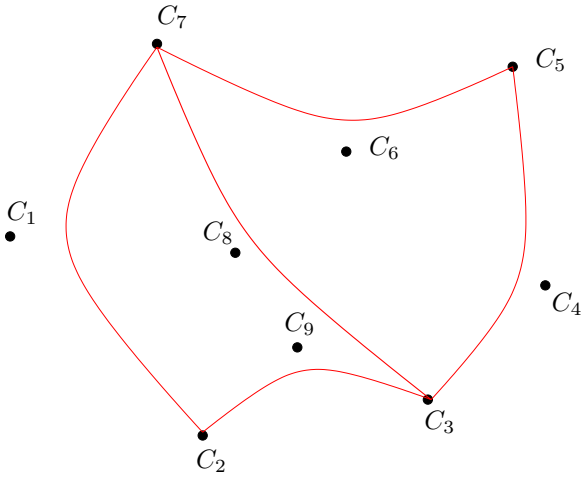
(a)
Initial $\mathbb{P}^1$ mesh



(b)
Initial subdivided $\mathbb{P}^1$ mesh

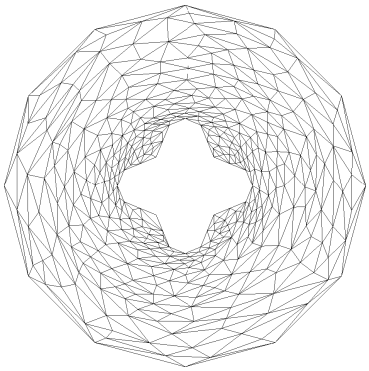


(c)
Deformed sudivided mesh

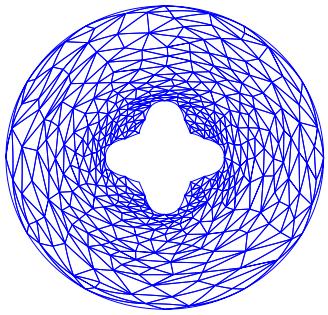


(d)
Final curved mesh

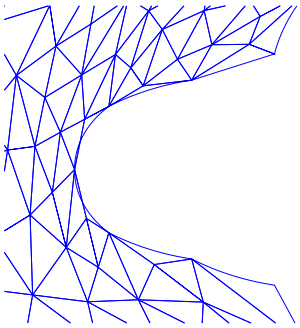
Figure 4: Illustration of the different steps of the algorithm. In red is plotted the boundary of Ω. It is parametrised by quadratic Bézier curves. In inital mesh is represented in (a). We introduce the mid points of the edges and refined the initial $\mathbb{P}^1$ mesh, see (b). This mesh is then deformed so that the control points that defines the curved boundary are exactly patched, see (c). Then the internal curved boundaries are computed using the control points, see (d).



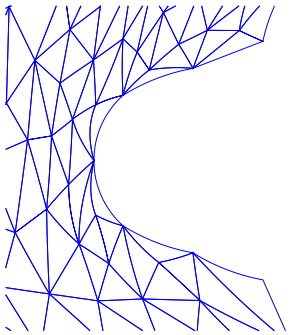
(a) $\mathbb{P}^1$ mesh.



(b) Valid curved mesh

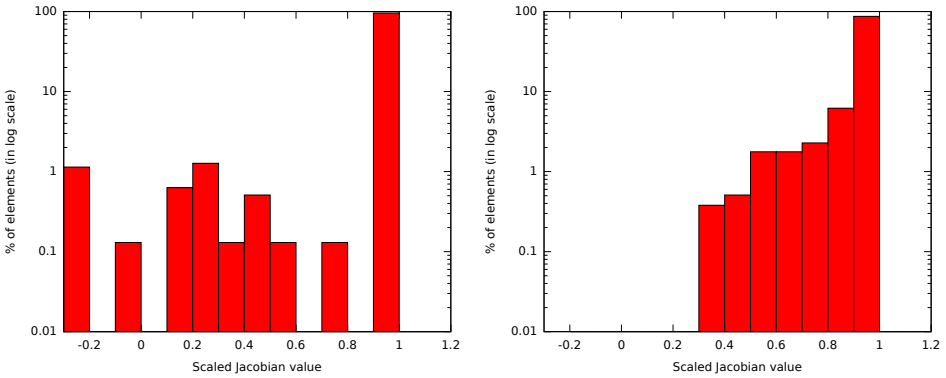


(c) Invalid elements.



(d) Zoom in the valid curved mesh.

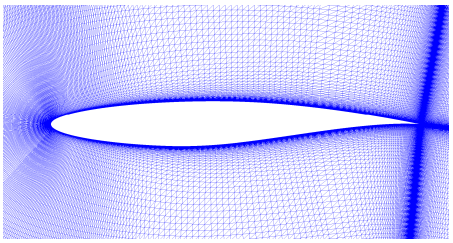
Figure 5: First isotropic test case



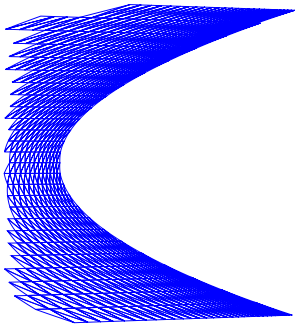
(a) Scaled Jacobians of init curved mesh.

(b) Scaled Jacobians of final curved mesh.

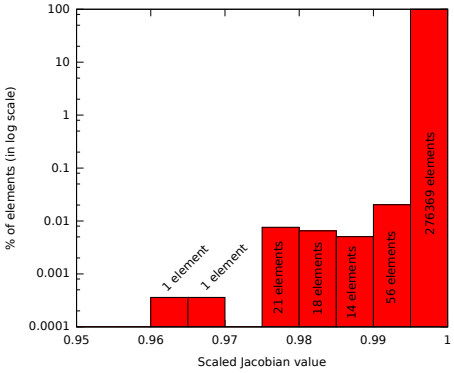
Figure 6: Isotropic test case.



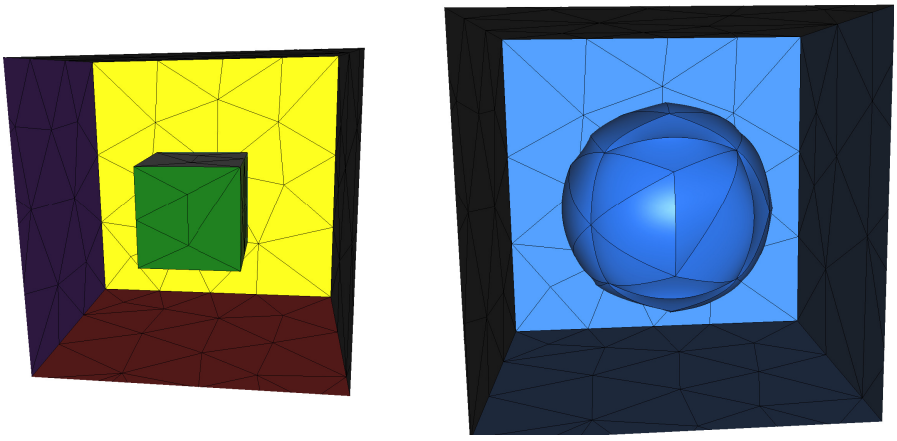
(a) Curved mesh.



(b) Zoom on curved mesh.

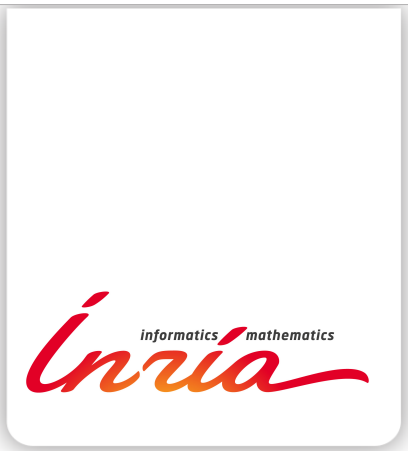


(c) Scaled Jacobians.



(d) Initial mesh, internal boundary (e) Final geometry, internal boundary

Figure 10: Initial and final geometry in 3D.



RESEARCH CENTRE
BORDEAUX – SUD-OUEST

351, Cours de la Libération
Bâtiment A 29
33405 Talence Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399