INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2014; **76**:246–266
Published online 12 July 2014 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/fld.3932

A method for computing curved meshes via the linear elasticity analogy, application to fluid dynamics problems

R. Abgrall[1,2,3,*,†], C. Dobrzynski[1,2,*,†] and A. Froehly[1,2]

[1]*Team Bacchus, INRIA, 200 chemin de la vieille Tour, 33405 CEDEX Talence, France*
[2]*IMB, Université de Bordeaux, 351 cours de la libération, 33405 CEDEX Talence France*
[3]*Institut für Mathematik, Universität Zürich, Winterthurerstrasse 190, CH-8057 Zürich, Switzerland*

SUMMARY

We propose and analyze an algorithm for the robust construction of curved meshes in two and three dimensions. The meshes are made of curved simplexes. The algorithm starts from a mesh made of straight simplexes, and using a linear elasticity analogy applied on well-chosen data, one can generate a curved mesh. Note that if the initial mesh has a boundary layer, this method allows to conserve it on the final mesh. This algorithm is used on several airfoils in two and three dimensions, including a turbulent M6 wing. Copyright © 2014 John Wiley & Sons, Ltd.

Received 19 March 2013; Revised 27 March 2014; Accepted 1 June 2014

KEY WORDS: curved meshes generation; triangulation; high-order schemes

1. INTRODUCTION

We are interested in the simulation of steady compressible fluid problems by mean of a formally high-order accurate scheme designed to use unstructured triangular/tetrahedra meshes. The problem is formalized by the Navier–Stokes equations (laminar or turbulent) written in a domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$. A part of the boundary of Ω, that we denote $\partial\Omega_s$, corresponds to the solid boundary, and the rest, denoted by $\partial\Omega_f$, corresponds to the fluid boundary where typically, one applies inflow/outflow conditions. On $\partial\Omega_s$, the velocity is set to 0, and in this paper, we assume adiabatic wall.

Hence, the goal is to discretize the steady Navier–Stokes equations, which describe the evolution of $W = (\rho, \rho\mathbf{u}, E, \mu_t)^T$, where ρ is the density, $\mathbf{u}$ is the velocity, and if ε is the specific internal energy, $E = \rho\varepsilon + \frac{1}{2}\rho\mathbf{u}^2$. The variable η describes the variables needed for the turbulent model. Here, we assume Spalart–Allamas model, so that μ_t is the turbulent viscosity

$$\operatorname{div}(F_e(W) - F_v(W, \nabla W)) = S_{turb} \qquad \text{in } \Omega \tag{1a}$$

subjected to boundary conditions

$$\mathbf{u} = 0 \text{ and } \nabla T \cdot \mathbf{n} = 0 \qquad \text{on } \partial\Omega_s \tag{1b}$$

and inflow/outflow boundary conditions

$$W = W_\infty \text{ on } \partial\Omega_\infty. \tag{1c}$$

*Correspondence to: R. Abgrall and C. Dobrzynski, Institut für Mathematik, Universität Zürich, Winterthurerstrasse 190, CH-8057 Zürich, Switzerland.
†E-mail: remi.abgrall@math.uzh.ch, cecile.dobrzynski@math.u-bordeaux1.fr

In (1a), S_{turb} is the source term for Spalart–Allmaras model. In (1b), $\mathbf{n}$ is the inward normal vector, Ω_s is the solid boundary, and Ω_∞ is the fluid boundary, see Figure 1.

From the geometrical point of view, we are given a domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, in which boundary can be parametrized by means of Bézier or non-uniform rational B-spline (NURBS) patches. In this work, we assume that the patches are triangular (in 3D) or simple segments, see Figure 2, in 2D.

A huge amount of work by many researchers throughout the world is being done to design efficient and robust algorithms for fluid problems with accuracy formally higher than the second order. If we specialize to the compressible fluid problems only, examples are given by the DG methods, see for example, among many other, [1–3], the finite element-like methods such as the Streamline Upwind Petrok Galerkin (SUPG) method [4, 5], the isogeometric analysis method [6], or the residual distribution methods [7, 8]. In this paper, we are interested in the residual distribution schemes.

However, it is known, because the seminal work by Bassi and Rebay [1], that the true accuracy of a high-order method can only be revealed provided that the boundary is represented with at least the same accuracy. This indicates that much effort is needed to be put on the generation of meshes that fit well the boundaries, and *this is a central point of the algorithm design.*

This paper is mainly a contribution to provide an efficient and robust algorithm that is able to construct, in 2D and 3D, curved meshes that fit exactly to a boundary, which is defined by mean of quadratic Bézier or NURBS patches. We are not interested in discussing in full details the numerical algorithm that we use to compute the numerical solution nor the structure of the fluid model, our main interest is in the generation of curved meshes that fit exactly a given curved solid boundary. We restrict ourselves to elements that are logical simplexes.

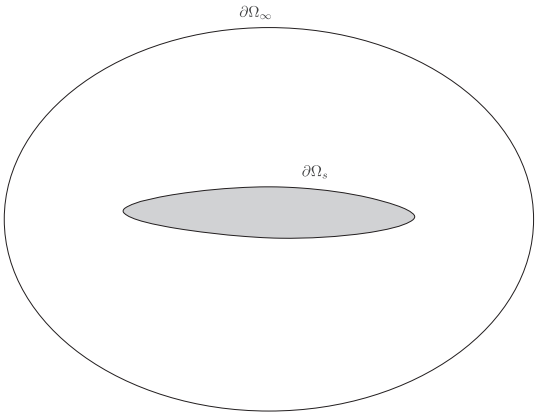


Figure 1. Simplified geometry for the boundary condition description.

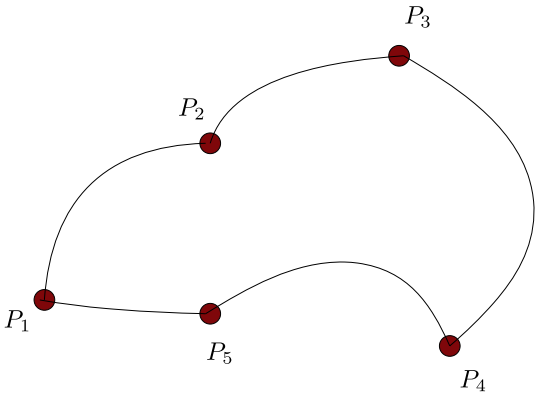


Figure 2. Example of a parametrized domain. For any $[P_i, P_{i+1}]$, the curve is parametrized by a Bézier or a NURBS curve.

DOI: 10.1002/fld

In fact, the construction of meshes by triangles and tetrahedra, while respecting the boundaries, is a well-developed topic, see e.g. [9, 10]. However, the construction of curved meshes is not a trivial matter, especially in the case of boundary layers and nonconvex domains. Indeed, when the boundary of the domain is described with curves or surfaces and if the associated boundary mesh is also composed of curves, the volumic mesh of the domain needs to be curved to obtain a valid mesh (that means a mesh without crossing edges). A way to deal with this problem is to start from a triangles/tetrahedron mesh with curved elements on the boundary and to search to make the volumic mesh valid. There are several ways to do that: perform some local modifications on the volumic mesh [10–14], propagate the curvature inside the domain with smoothing schemes such as Laplacian, linear elasticity smoothing [15, 16], nonlinear elasticity smoothing [17], or use an optimization algorithm [18].

The objective of this paper is to propose an entirely automatic and robust algorithm that is able to construct valid curved meshes even in the case of boundary layer. 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 on the quadratic case, but the algorithm can be, *a priory*, extended to any order.

We first start by describing quickly, because this is not the central point of the paper, the numerical scheme we are going to use for the CFD applications. Then, we recall a few classical information on Bézier and NURBS approximation then we describe the algorithm and analyze it. Finally, we provide examples in two and three dimensions. We end by applying the method on two turbulent airfoil configurations, in order to show that these meshes are suitable for CFD simulations.

2. A NUMERICAL SCHEME

We briefly sketch the numerical method. The details can de found in [19, 20].

Let us consider a tessellation $\mathcal{T}_h$ of Ω with (curved) triangles or tetrahedra. We assume a conformal approximation W^h by quadratic element. We denote by K a generic element and σ a generic DOF. For each element K of $\mathcal{T}_h$, the principle of the method is the following:

1. Compute the total residual

$$\Phi_K = \int_{\partial K} \left(F_e(W^h) - F_v\left(W^h, \overline{\nabla W^h}\right) - S_{Turb} \right) \right),$$

   where, following [19], a continuous approximation of the gradient is evaluated. The best strategy is using Zienckewicz and Zu optimal patch recovery, see [19]. The integral is evaluated thanks to Gaussian formula incorporating the fact that the surface of the element is not planar. This is basically the only place where the geometry of the element is needed.
2. Evaluate for each DOF σ in K, a monotone residual $\Phi_\sigma^{M,K}$. They must satisfy

$$\sum_{\sigma \in K} \Phi_\sigma^{M,K} = \Phi_K.$$

   The description of $\Phi_{\sigma,K}^H$ (that boils down to Lax–Friedrichs when the viscous and turbulent terms cancel) is not necessary for the sake of this paper, see again [19–21].
3. We evaluate the high-order residuals $\Phi_{\sigma,K}^H$, which also satisfy the conservation relation

$$\sum_{\sigma \in K} \Phi_{\sigma,K}^H = \Phi_K.$$

   The evaluation of $\Phi_{\sigma,K}^H$ is done such as in the inviscid limit, the solution is non-oscillatory, see [22]. This can be achieved thanks to the knowledge of $\left\{\Phi_{\sigma,K}^M\right\}_{\sigma \in K}$.

   For steady problems, the high-order residuals applied to the interpolation W_{app}^h of a smooth solution satisfy $\Phi_{\sigma,K}^H\left(W_{app}^h\right) = O(h^{2+d})$ to guaranty that the scheme is third-order accurate.

DOI: 10.1002/fld

Similarly, for each boundary element Γ, one can define the total boundary residual and high-order boundary $\left\{\Phi^H_{\sigma,\Gamma}\right\}_{\alpha\in\Gamma}$ residuals that enable the weak boundary conditions (1b) and (1c). Again, Gaussian quadrature formula is used. The details are not useful for our discussion, see [20, 21] again.

Once this is done, the numerical solution is asked to satisfy for any σ

$$\begin{aligned} &\sum_{K,\sigma\in K} \Phi^H_{\sigma,K} = 0 && \text{if } \sigma \notin \partial\Omega, \\ &\sum_{K,\sigma\in K} \Phi^H_{\sigma,K} + \sum_{\Gamma,\sigma\in\Gamma} \Phi^H_{\sigma,\Gamma} = 0 && \text{else.} \end{aligned} \tag{2}$$

The system (2) is solved by a nonlinear LUSGS algorithm.

3. BÉZIER AND NURBS APPROXIMATIONS

Throughout the paper, we only consider Bézier and NURBS approximations with triangular patches. We recall their structure on simplexes, the case of linear, triangular, and tetrahedral elements are only particular cases.

Let us consider $K \subset \mathbb{R}^d$ a simplex, that is, the convex hull of $d+1$ independent points $\{P_1, \ldots, P_{d+1}\}$, that is,

$$\mathrm{Vol}(K) = \frac{1}{d+1}\left|\det\left(\overrightarrow{P_{d+1}P_1}, \ldots, \overrightarrow{P_{d+1}P_d}\right)\right| > 0. \tag{3}$$

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

$$\begin{aligned} \sum_{i=1}^{d+1} \lambda_i(M) &= 1 \\ M &= \sum_{i=1}^{d+1} \lambda_i(M) P_i. \end{aligned} \tag{4a}$$

These numbers are unique and $\lambda_i(M)$ are equal to the volume of the convex hull $\{P_1, \ldots, P_{i-1}, M, P_{i+1}, \ldots, P_{d+1}\}$ divided by the volume of the convex hull $\{P_1, \ldots, P_{d+1}\}$. This leads to formula (4b), where $j \neq i$ is arbitrary

$$\lambda_i(M) = \frac{\det\left(\overrightarrow{P_jP_1}, \ldots, \widehat{\overrightarrow{P_jP_i}}, \ldots, \overrightarrow{P_jP_d}\right)}{\det\left(\overrightarrow{P_jP_1}, \ldots, \overrightarrow{P_jP_d}\right)}, \tag{4b}$$

and for any j,

$$\widehat{\overrightarrow{P_jP_i}} = \overrightarrow{P_jM}. \tag{4c}$$

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

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

DOI: 10.1002/fld

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

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

The Bézier polynomial is

$$B_\alpha^n(M) := c(\alpha)\Pi_{i=1}^{d+1}\lambda_i(M)^{\alpha_i}. \tag{5}$$

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

$$\begin{cases} B_\alpha^n(M) \geqslant 0 \text{ for any } \alpha, |\alpha| = n \text{ and } M \in K \\ \displaystyle\sum_{\alpha,|\alpha|=n} B_\alpha^n(M) = 1. \end{cases} \tag{6}$$

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

$$N_\alpha^n = \frac{\omega_\alpha B_\alpha^n}{\displaystyle\sum_{\alpha',|\alpha'|=n} \omega_{\alpha'} B_{\alpha'}^n}. \tag{7}$$

Again,

$$\begin{cases} N_\alpha^n(M) \geqslant 0 \text{ for any } \alpha, |\alpha| = n \text{ and } M \in K \\ \displaystyle\sum_{\alpha,|\alpha|=n} N_\alpha^n(M) = 1 \end{cases} \tag{8}$$

In the following, we denote by φ_α^n either the functions B_α^n or N_α^n. They satisfy the relations (6) or (8)

$$\begin{cases} \varphi_\alpha^n(M) \geqslant 0 \text{ for any } \alpha, |\alpha| = n \text{ and } M \in K \\ \displaystyle\sum_{\alpha,|\alpha|=n} \varphi_\alpha^n(M) = 1. \end{cases} \tag{9}$$

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

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

Of course, the control parameters have to be chosen carefully for the approximation to be meaningful.

DOI: 10.1002/fld

We immediately get the following property

Property 3.1 (Convex hull property)
For any $M \in K$, the relations (9) imply 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.

Property 3.2
If K is a segment, the tangent vectors at the ends of the curve $\psi(K)$ have the same direction as the first and the last control polygon spans, respectively.

4. DESCRIPTION AND ANALYSIS OF THE ALGORITHM

Let us further investigate the convex hull Property 3.1 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 3.

On this figure, we have represented the points of barycentric coordinate $\alpha = \left(\frac{\alpha_1}{n}, \cdots, \frac{\alpha_d}{n}\right)$ 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

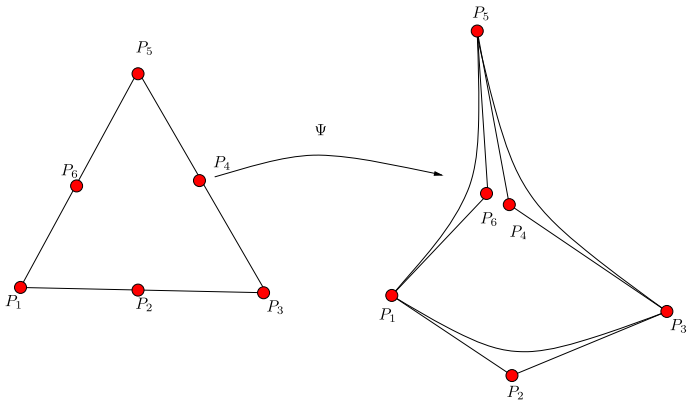


Figure 3. Transformation of K into $\hat{K}$ for a quadratic Bézier. The control points are represented in red.

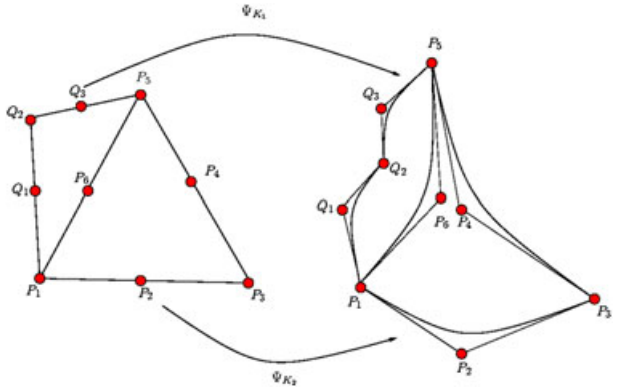


Figure 4. Transformation of two simplexes or a quadratic Bézier. The control points are represented in red.

DOI: 10.1002/fld

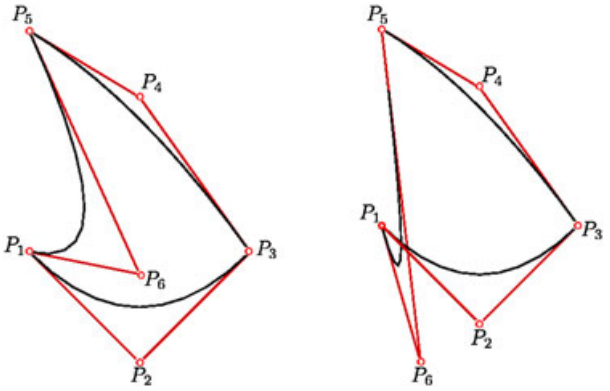


Figure 5. Curved triangles: on the left, the triangle is valid; on the right, the triangle is invalid.

Property 3.1. Considering, for example, the Figure 4, where two simplexes are represented, we see that under the same assumption, the two element images of mapping ψ_{K_1} and ψ_{K_2} will not intersect, because the image of a face is a face and the mapping continuous.

This condition, although valid, is very restrictive. In Figure 5, this condition says that the left triangle is invalid even if none of its boundary edges crossed. Indeed, the convex hull of the curve P_1P_2 intersects the convex hull of the curve P_1P_3.

It is possible to define a less restrictive condition based on the convex hull property and the property of the curve/surface tangency at its control polygon (Property 3.2). So on each vertex of the simplex, if we consider the angle between the two tangents at the boundary faces, we know that if this angle is positive, the boundaries don't cross. This criterion is less restrictive and more easy to verify. It is enough to check the orientation of the subsimplex composed by the vertex and the two adjacent control points. For example, in Figure 5, we have to check the triangles $P_1P_7P_9$, $P_7P_2P_8$, and $P_8P_3P_6$. That leads to conclude that the left triangle is valid and that the right triangle is not valid (the area of the triangle $P_1P_7P_9$ is negative). This remark is at the core of our method.

Remark 4.1 (about the Jacobian of the transformation)
The previous validity criterion does not imply that the mapping between K and $\hat{K}$ is unique. Thus, it is possible to have no crossing edges and that the Jacobian of the transformation will be negative somewhere on the triangle. To avoid this problem, it is possible to check an additional criterion on the Jacobian. Indeed, if we calculate the Jacobian of the mapping using the barycentric coordinates Λ_i, we find

$$\begin{aligned} J = {} & (\det(\mathbf{a}_{16},\mathbf{a}_{34}) + \det(\mathbf{a}_{24},\mathbf{a}_{26}))\Lambda_1\Lambda_2 + \det(\mathbf{a}_{16},\mathbf{a}_{26})\Lambda_1^2 \\ & - (\det(\mathbf{a}_{16},\mathbf{a}_{54}) + \det(\mathbf{a}_{56},\mathbf{a}_{26}))\Lambda_1\Lambda_3 + \det(\mathbf{a}_{24},\mathbf{a}_{34})\Lambda_2^2 \\ & - (\det(\mathbf{a}_{24},\mathbf{a}_{54}) + \det(\mathbf{a}_{56},\mathbf{a}_{34}))\Lambda_3\Lambda_2 + \det(\mathbf{a}_{56},\mathbf{a}_{54})\Lambda_3^2, \end{aligned}$$

where we have set $\mathbf{a}_{ij} = 2(P_i - P_j)$, it is a vector.

Hence, a sufficient condition for the curved element to be valid is

$$\begin{array}{lll} \det(\mathbf{a}_{16},\mathbf{a}_{26}) \geqslant 0 & \det(\mathbf{a}_{24},\mathbf{a}_{34}) \geqslant 0 & \det(\mathbf{a}_{56},\mathbf{a}_{54}) \geqslant 0 \\ \det(\mathbf{a}_{16},\mathbf{a}_{34}) + \det(\mathbf{a}_{24},\mathbf{a}_{26}) \geqslant 0 & \det(\mathbf{a}_{54},\mathbf{a}_{16}) + \det(\mathbf{a}_{26},\mathbf{a}_{56}) \geqslant 0 & \det(\mathbf{a}_{54},\mathbf{a}_{24}) + \det(\mathbf{a}_{34},\mathbf{a}_{56}) \geqslant 0, \end{array} \quad (10)$$

with at least one of these terms strictly positive.

DOI: 10.1002/fld

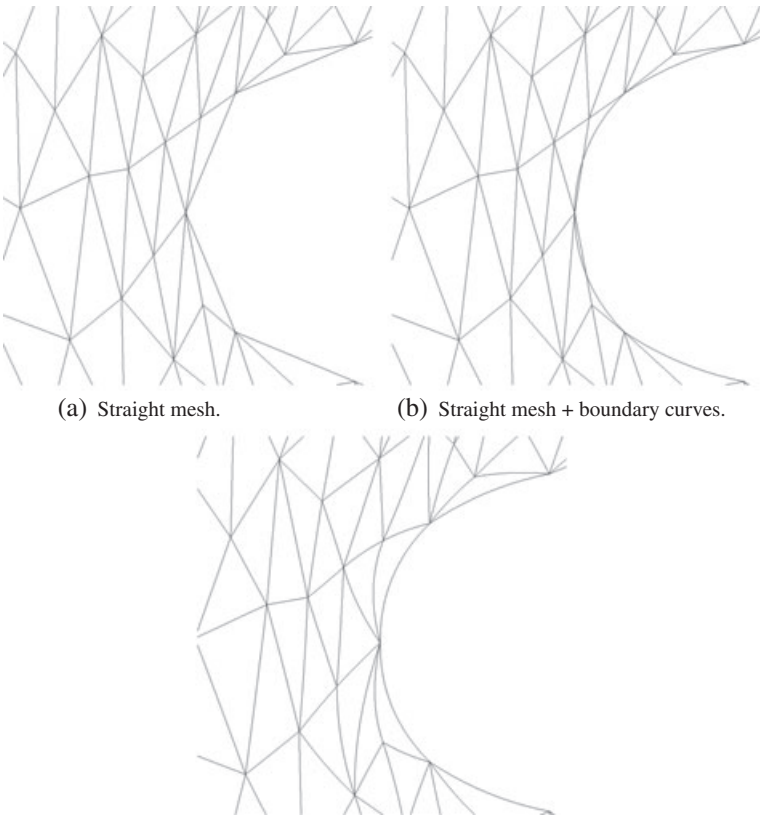
One can notice that the three first conditions are equivalent to compute the sub-area associated to the triangle vertex. Thus, to check uniqueness of the mapping, we only need to verify three additional conditions.

The same reasoning can be done in three dimensions that lead to verify the sub-volumes associated to the vertex and 16 others conditions.

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 `GMSH` [23] and `MMG3D` [24]. The idea is to deform this linear mesh. If one starts to deform only the linear boundary elements to produce curved boundary elements corresponding to the parametrized geometry, we may run into problems because curved faces may intersect volumic elements. An example of such problem is shown in Figure 6: on the left, a portion of the straight mesh is drawn; on the middle, the figure represents the same mesh with a curved representation of its boundary. One can see that the curvature of the boundary edges leads to crossing edges, this mesh is invalid.

So starting from the linear mesh and the parametrized geometry, we want to propagate the boundary curvature into the volumic mesh. To do that easily, we work on a subdivided mesh in which the vertexes are considered as *the control point* of our curved mesh. Thanks to Property 3.1 and 3.2, we know that if this subdivided mesh is valid, then the associated curved mesh will be also valid (no crossing edge). The main advantage to consider this subdivided mesh is that we only need to deal with linear elements: the volume evaluation of elements is very easy, hence, it is easy to know if this mesh is valid or not. Anticipating a bit the rest of this paper, the second reason is that we have to deal only with $\mathbb{P}^1$ finite element (in a linear elasticity solver), instead of $\mathbb{P}^2$, quadratic Bézier



(a) Straight mesh. (b) Straight mesh + boundary curves.

(c) Valid curved mesh.

Figure 6. An example of invalid curved mesh. (a) Straight mesh, (b) straight mesh + boundary curves, and (c) valid curved mesh.

DOI: 10.1002/fld

or NURBS into a linear elasticity solver, this is much cheaper and we have a better control on the individual behaviors of the control points.

In other words, instead of deforming the linear elements, we first subdivide them by adding the midpoints of edges, more generally the Lagrange points of the elements. Here, we interpret them as control points. Then, we subdivide the elements to obtain the *subdivided* mesh, see Figure 7, and we deform it. 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 3.1. In Figure 6(c), a valid curved mesh is represented.

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

1. The boundary points of this mesh match to the control points defining the curved surface.
2. All the elements of the deformed mesh are legal.

One way of achieving this goal is 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

$$\begin{aligned} \operatorname{div}\left(\lambda \operatorname{tr}(\nabla u) + \mu(\nabla u + \nabla u^T)\right) &= 0 \text{ on } \Omega \\ u &= g \text{ on } \partial\Omega. \end{aligned} \tag{11}$$

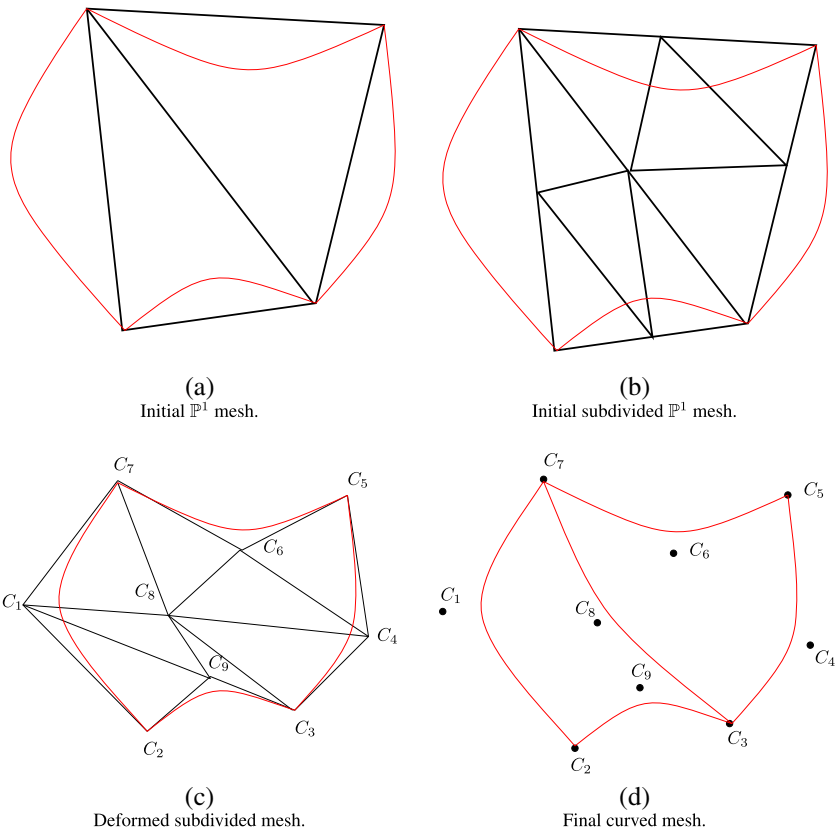


Figure 7. Illustration of the different steps of the algorithm. In red is plotted the boundary of Ω. It is parametrized by quadratic Bézier curves. An initial mesh is represented in (a). We introduce the mid points of the edges and refined the initial $\mathbb{P}^1$ mesh, see (b). This mesh is then deformed so that the control points that define the curved boundary are exactly matched, see (c). Then, the internal curved boundaries are computed using the control points, see (d). (a) Initial $\mathbb{P}^1$ mesh; (b) initial subdivided $\mathbb{P}^1$ mesh; (c) deformed subdivided mesh, and (d) final curved mesh.

DOI: 10.1002/fld

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

$$M^D \text{ vertex of } \mathcal{T}_h^D \text{ if and only if there exists (a unique) } M^0 \in \mathcal{T}_h^0 \text{ such that } M^D = M^0 + u(M^0). \tag{12}$$

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

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

Unfortunately, the deformed mesh may not be legal if the deformation is too large. To overcome this problem, we notice that the problem (11) 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 $K^0 = \text{convex}\left(A^0_{i_1}, \ldots, A^0_{i_{d+1}}\right)$ of $\mathcal{T}_h^0$ the set

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

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

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

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

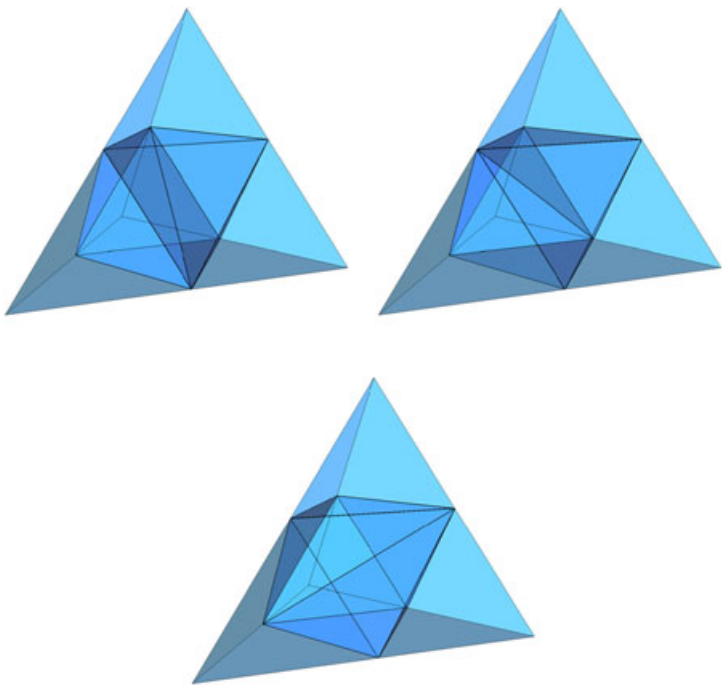


Figure 8. The three decompositions for the tetrahedron.

DOI: 10.1002/fld

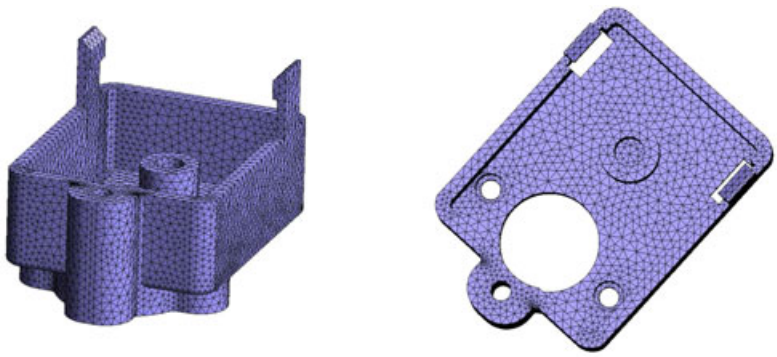
For obvious numerical reasons, we cannot accept elements with zero volume. It is necessary to strengthen a bit the previous criteria; if ω_{min} is the smallest volume of elements on the initial mesh, we impose $\omega_{K^0}(\theta) > \lambda\ \omega_{min}$. In practise, we have chosen $\lambda = 0.9$.

The algorithm is thus the following:

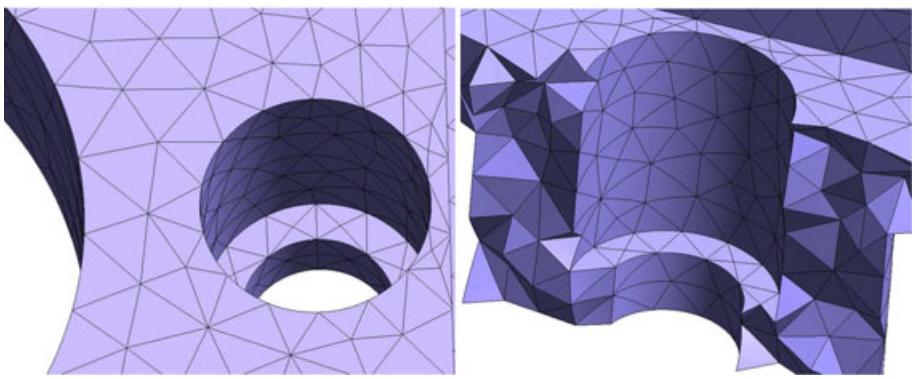
0. Create the subdivided mesh $\mathcal{T}^0$.
1. Solve for g in $\mathcal{T}^0$.
2. Look for the smallest θ, say θ_0, such that for any $\theta \in [0, \theta_0[$, $\omega_{K^0}(\theta) > \lambda\ \omega_{min}$ for any K^0 in $\mathcal{T}^0$, then
   - If $\theta_0 > 1$, the final mesh is obtained, go to step 3.
   - If $\theta_0 < 1$, then update the mesh following (12), denote this new mesh as $\mathcal{T}^0$ and go to step 1.
3. Recompose the curved mesh from the final $\mathcal{T}^0$.

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.

Remark 4.2 (About the creation of the subdivided mesh)
If the subdivision of the $\mathbb{P}^1$ mesh by adding the Lagrange points is unique on dimension two, this is not true on dimension three. Indeed, each tetrahedron is decomposed on eight tetrahedra, four of them are composed by one vertex of the $\mathbb{P}^1$ tetrahedron and the middle of its adjacency edges; for creating the four others, there are three possible configurations, see Figure 8. During the first step of the algorithm, we have to make a choice and we decide for each tetrahedron to take the configuration composed with the tetrahedra having the maximum volume. In severe configurations



(a) Curved surface mesh.



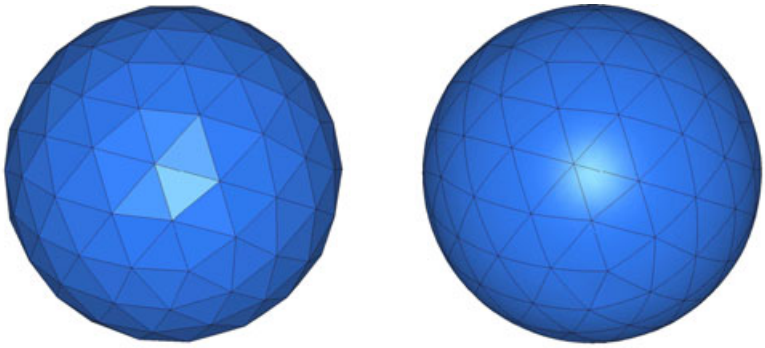
(b) On the left: zoom on the curved surface, on the right: volumic cut on the tetrahedron.

Figure 9. Meshes for a mechanical piece. (a) Curved surface mesh and (b) on the left: zoom on the curved surface, on the right: volumic cut on the tetrahedron.

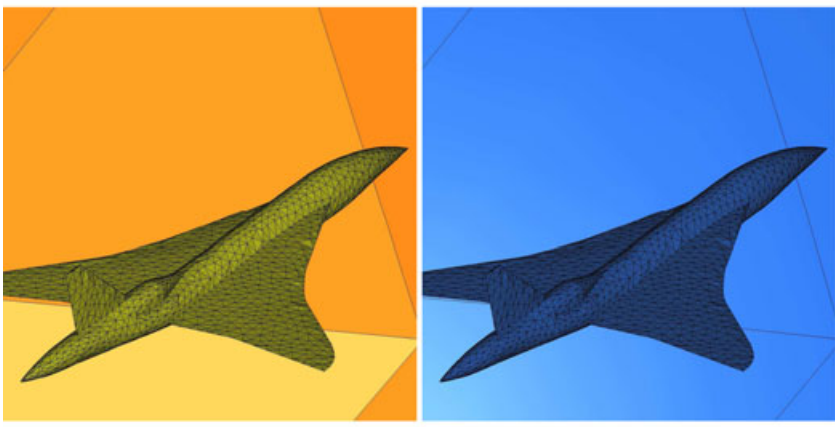
DOI: 10.1002/fld

(such as turbulent boundary layers), we may use this variety of choice during the simulation to facilitate the convergence.

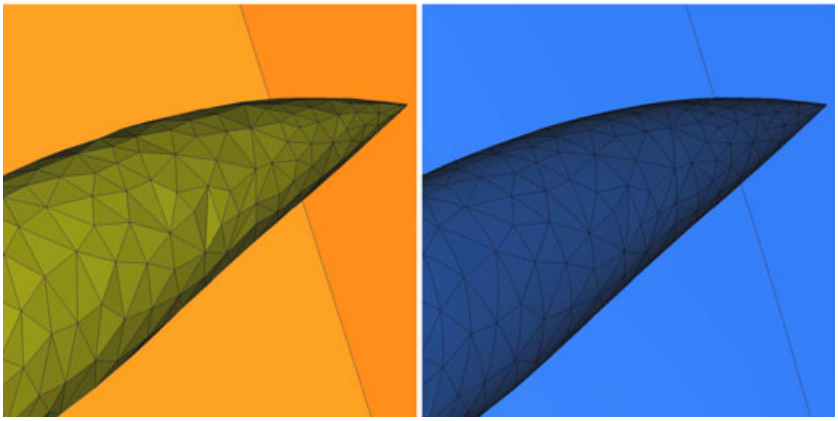
Remark 4.3 (About mapping uniqueness)
To ensure that the mapping is unique, one can add to the previous algorithm a step 2, which consists to find the smallest θ_{jac} such as the Jacobian becomes negative and to update the mesh of min (θ_0, θ_{jac}).



(a) P1 mesh ; external curved surface.



(b) Airplane.



(c) Zoom on the airplane.

Figure 10. Meshes for airplane test case. (a) P1 mesh; external curved surface; (b) airplane, and (c) zoom on the airplane.

DOI: 10.1002/fld

5. EXAMPLES

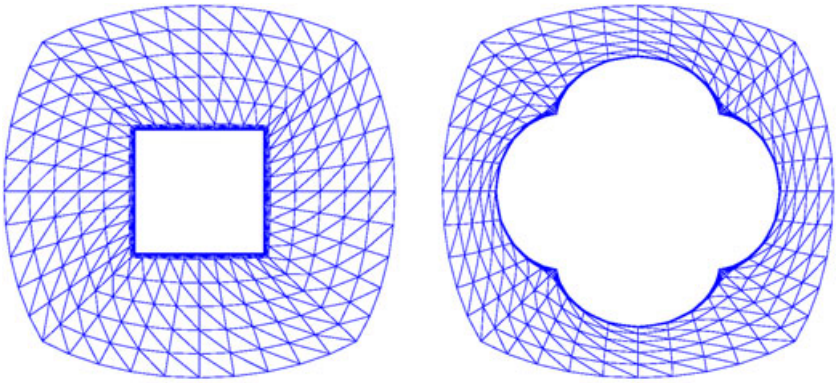
Many examples on two and three dimensions have been performed. In this paragraph, we explain some of them to demonstrate the potential of our method.

In this work, we use S_J, the scaled Jacobian measure defined in [13] to evaluate if an element is valid or not. As shown in [18], this measure does not allow to ensure that a curved element has a nice shape or not but it is commonly used to detect the invalid elements.

We note ξ the coordinates in the parametric space, E an element of this space and $T(\xi)$ the transformation 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

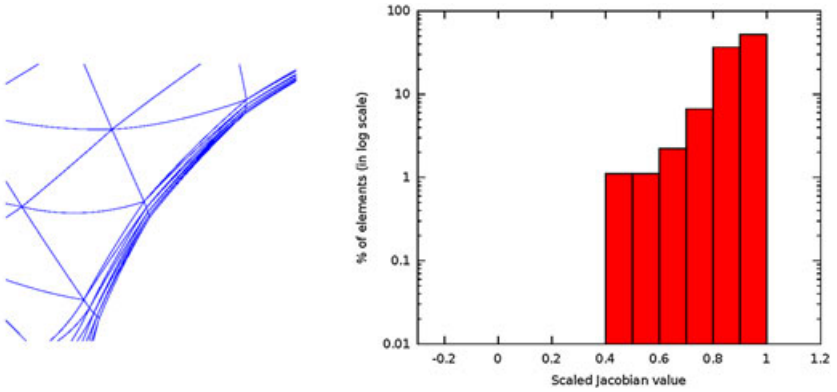
$$S_J = \frac{\min\limits_{\xi \in E} |J_T(\xi)|}{\max\limits_{\xi \in E} |J_T(\xi)|}. \tag{13}$$

Because the Jacobian of the transformation is constant for a straight element, $S_J = 1$ in this case. 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).



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

Figure 11. Boundary layers test case. (a) Initial straight mesh and (b) final curved mesh.



(a) Zoom on boundary layer test case. (b) Scaled Jacobians of last curved mesh.

Figure 12. Analytical boundary layers test case. (a) Zoom on boundary layer test case and (b) scaled Jacobians of the last curved mesh.

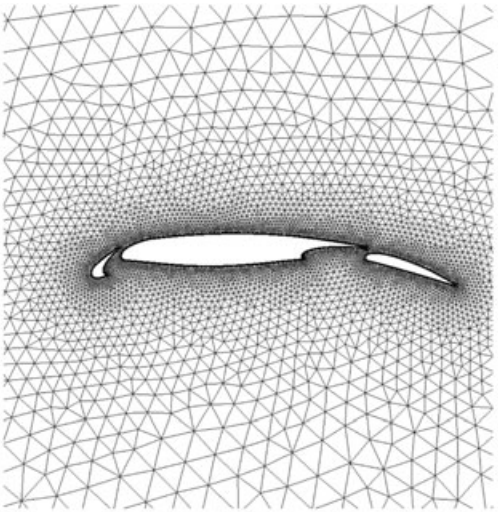
DOI: 10.1002/fld

5.1. *Isotropic curved mesh examples*

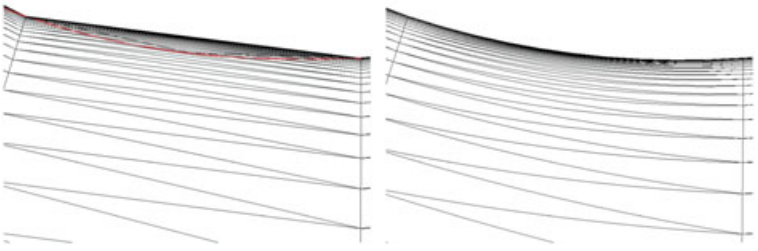
Most of the isotropic $\mathbb{P}^1$ mesh can certainly be transformed into a curved mesh without any problem if the deformation between the piecewise linear boundary and the curved one is not too large because the curvature of the boundary does not create invalid elements. However, using linear elasticity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

5.1.1. Mechanical part. In this example, we consider a mechanical part composed of several holes and cylinders. This mesh contains 60 606 vertexes, 14 240 triangles, and 50 665 tetrahedra. When the boundary is curved without modifying the volumic mesh, there are five wrong tetrahedra. The algorithm succeeds with two iterations. A volumic cut into the curved tetrahedron is shown in Figure 9.

5.1.2. Airplane. We consider an airplane in a large bounding box, see Figure 10. The initial mesh contains 77 061 vertexes and 443 458 tetrahedra, and when we curve the boundary mesh only, the volumic mesh contains one invalid element. For this test case, we need nine iterations of our algorithm to obtain a valid $\mathbb{P}^2$ mesh. The invalid element is almost a sliver: its volume is quasi null and its vertexes are quasi coplanar. The final $\mathbb{P}^2$ mesh is valid: the minimum scaled Jacobian is positive but we can remark that the *associated* $\mathbb{P}^1$ mesh is invalid (it contains elements with negative volume).



(a) Airfoil.



(b) Zoom on the straight mesh with curved boundary in red.

(c) Zoom on the final curved mesh.

Figure 13. Boundary layer-type mesh for an airfoil. (a) Airfoil; (b) zoom on the straight mesh with curved boundary in red, and (c) zoom on the final curved mesh.

DOI: 10.1002/fld

5.2. *Boundary layers and anisotropic meshes*

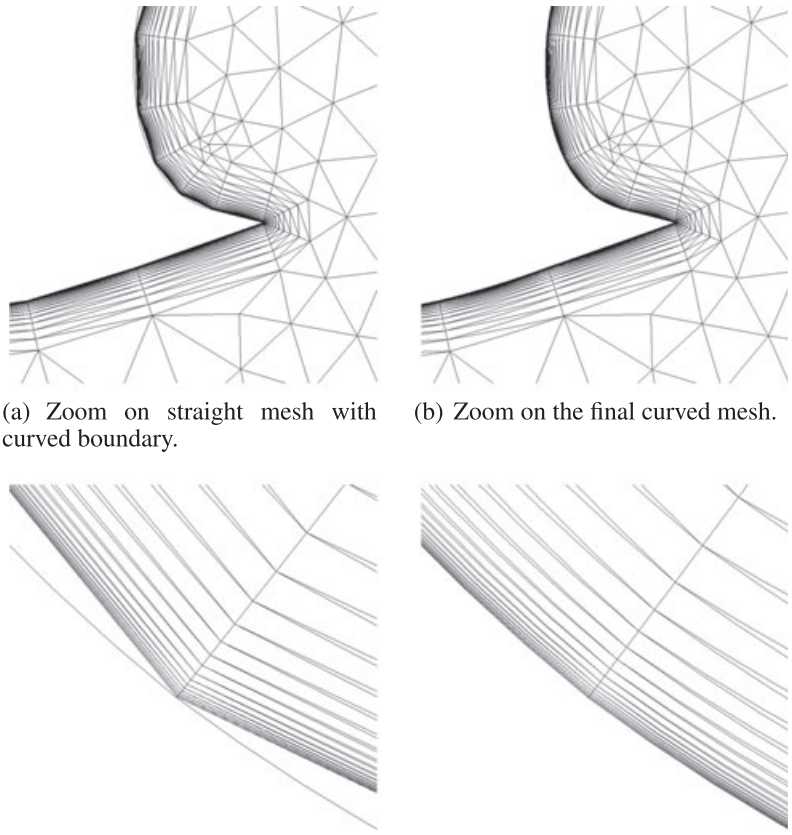
Let us consider now examples of mesh with boundary layers. This is *a priori* more challenging because we want to preserve the structure of the boundary layer mesh. The original mesh consists, near the boundary, of succession of layered cells. We want to keep this structure.

5.2.1. Analytical example. In this example, we apply a large deformation on a very stretch mesh. The initial mesh is represented in Figure 11(a). It has 400 vertexes and 720 triangles. We want to map the internal square onto a clover-like shape, see Figure 11(b). Obviously, the deformation is so large that one cannot only deform the boundary.

The mesh represented in Figure 11(b) is computed with our algorithm. We need six iterations only to obtain a valid curved mesh representing exactly the clover-like shape. All the elements are valid and most of them have a scaled Jacobian larger than 0.9, a total of 704 elements over the 720 elements of the mesh, see Figure 12(b).

Figure 12(a) shows a zoom in a part of Figure 11(b) to see that the structure of the initial mesh is conserved.

5.2.2. Three-component wing. In this example, we generate a boundary layer-type mesh for a three-component wing with GMSH. The mesh size normal to the wall is 0.005, the mesh size tangent to the wall is 3 (hence, the aspect ratio near the boundary is 600) and the bounding box of the domain is $[-4615.2, 5384.8] \times [-4985.3, 5014.7]$. Hence, the mesh is very stretched along the wing, see



(a) Zoom on straight mesh with curved boundary.

(b) Zoom on the final curved mesh.

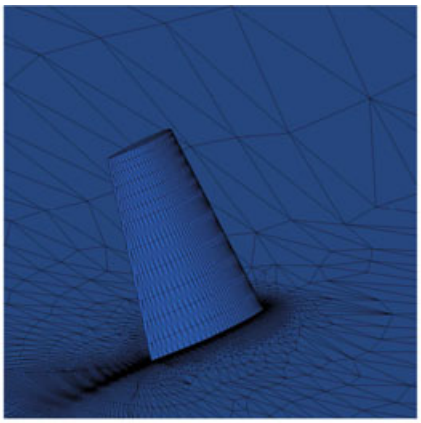
(c) Zoom on straight mesh with curved boundary.

(d) Zoom on the final curved mesh.

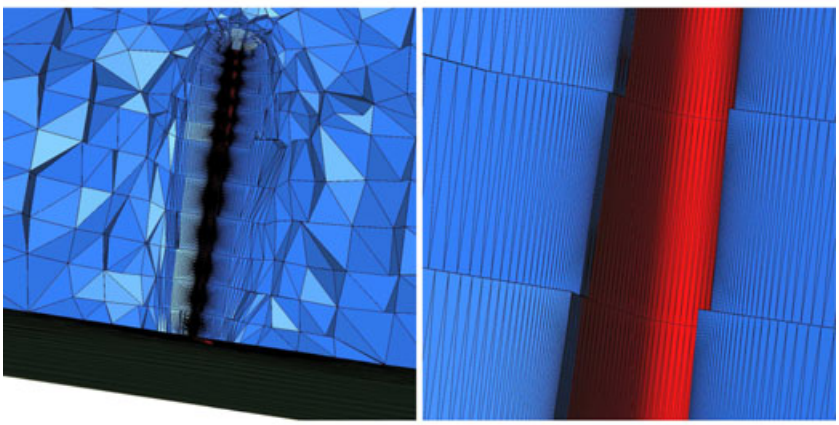
Figure 14. Boundary layer-type mesh for an airfoil: zoom. (a) Zoom on straight mesh with curved boundary; (b) zoom on the final curved mesh; (c) zoom on straight mesh with curved boundary, and (d) zoom on the final curved mesh.

DOI: 10.1002/fld

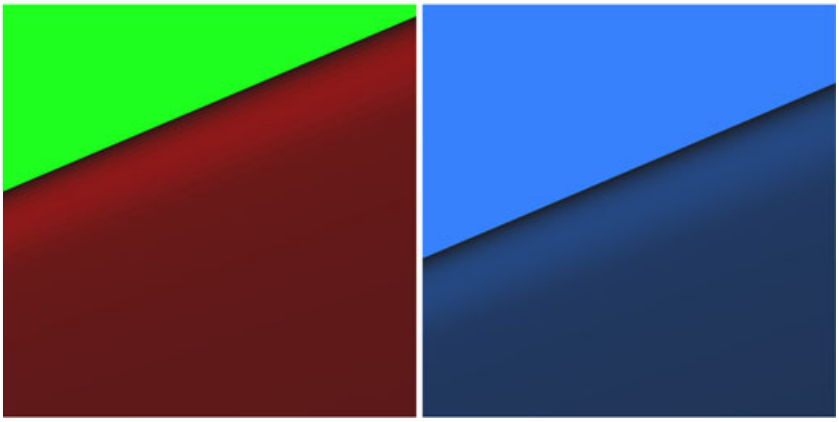
Figure 13. The mesh is composed by 41 391 vertexes and 20 023 triangles. In Figure 13(b), we can see that the curved boundary crosses many elements because of their thickness near the wall. The final curved mesh is obtained with our algorithm with one iteration. We can see (Figure 13(c)) that the mesh is valid and the structure of the boundary layers is conserved. Moreover, even in the zone where the mesh was valid, such as in Figure 14, the resolution of the linear elasticity system improves the mesh, the thickness of the boundary layer is the same as in the initial mesh.



(a) Mesh of the wing : P1 mesh.



(b) Zoom on the volumic mesh around the wing.



(c) Zoom on the wing. On the left: P1 mesh, on the right: curved surface.

Figure 15. Meshes for Dassault wing test case. (a) Mesh of the wing: P1 mesh; (b) zoom on the volumic mesh around the wing, and (c) zoom on the wing. On the left: P1 mesh, on the right: curved surface.

DOI: 10.1002/fld

5.2.3. M6 Wing. We consider an M6 wing meshed with boundary layers. The initial straight mesh contains 77 061 vertexes, 13 776 triangles, and 443 458 tetrahedra. This mesh is originally designed for the turbulent simulation of Section 6.2, it is very fine and very stretched around the wing (see Figure 15). There are many invalid elements (more than 100) when we try to curve the surface mesh only. Our strategy of elasticity analogy succeeds in only one shoot. In Figure 15, we give some snapshots of the initial straight mesh and the final curved mesh in order to see the structure of the mesh.

6. APPLICATION TO TURBULENT FLOW PROBLEMS

Several examples have been performed in two and three dimensions using the numerical scheme of Section 2.

6.1. RAE2822 case

This is the classical RAE 2822 case. The free stream Mach number is $M = 0.734$, the angle of attack is $2.79°$, and the Reynolds number is 6.5×10^6. Again, we start from a $\mathbb{P}^1$ mesh, apply our algorithm, and obtain the mesh displayed in Figure 16. The grid has 161 192 elements. The Mach number is shown in Figure 17. Figure 18 shows a comparison, for the pressure coefficient, between

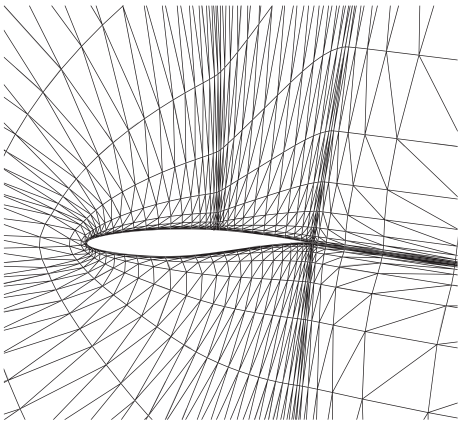


Figure 16. Quadratic mesh for the RAE2822 problem.

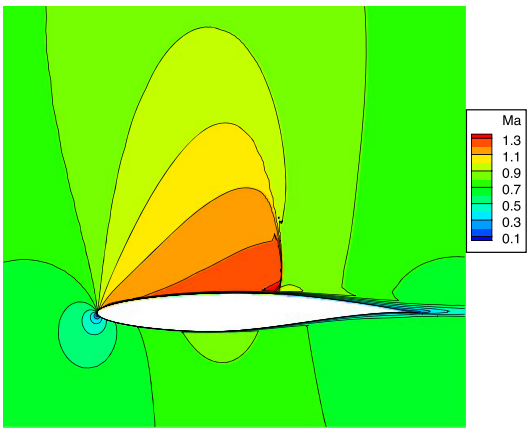


Figure 17. Mach number distribution, RAE2822 problem.

DOI: 10.1002/fld

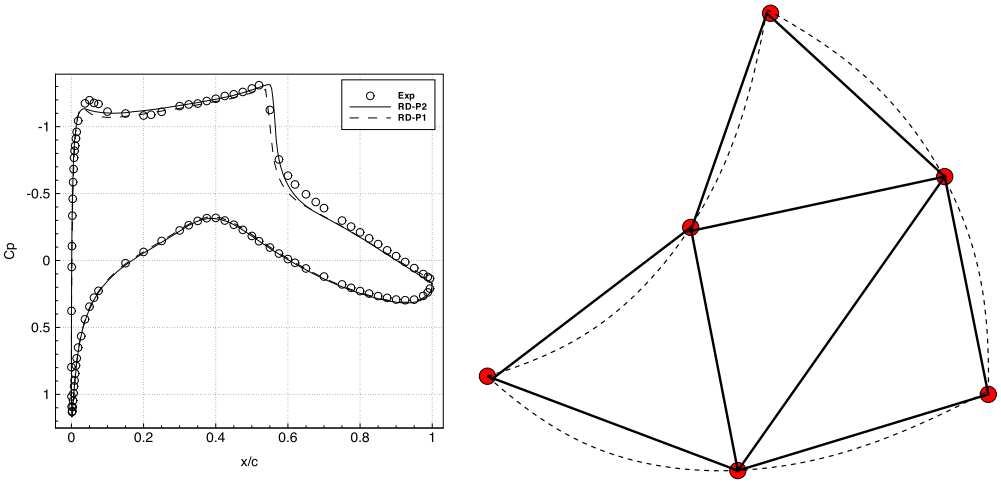


Figure 18. (a) Pressure coefficient, comparison between experimental data, results obtained with a $\mathbb{P}^1$ mesh and the quadratic mesh; (b) split of a $\mathbb{P}^2$ element (dotted lines) into four $\mathbb{P}^1$ linear elements.

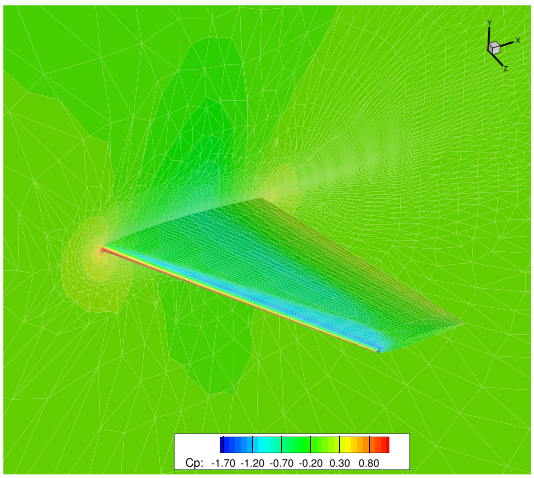


Figure 19. C_p coefficient.

experimental results, results obtained by subdividing the quadratic elements into linear ones (see Figure 18(b)) and this quadratic mesh.

6.2. *ONERA six-wing case*

We consider the M6 wing described in Section 5.2.3. The numerical scheme sketched in the previous paragraph is used for solving the following turbulent case:

- Inflow mach number: $M_\infty = 0.8395$,
- Angle of Attack: $\alpha = 3.06°$,
- Reynolds number: $\mathrm{Re} = 11.72 \times 10^6$.

The pressure coefficient C_p over the wing is displayed in Figure 19. The cross section of the pressure coefficients at different locations on the wing is displayed in Figure 20 and compared with the experimental results.

DOI: 10.1002/fld

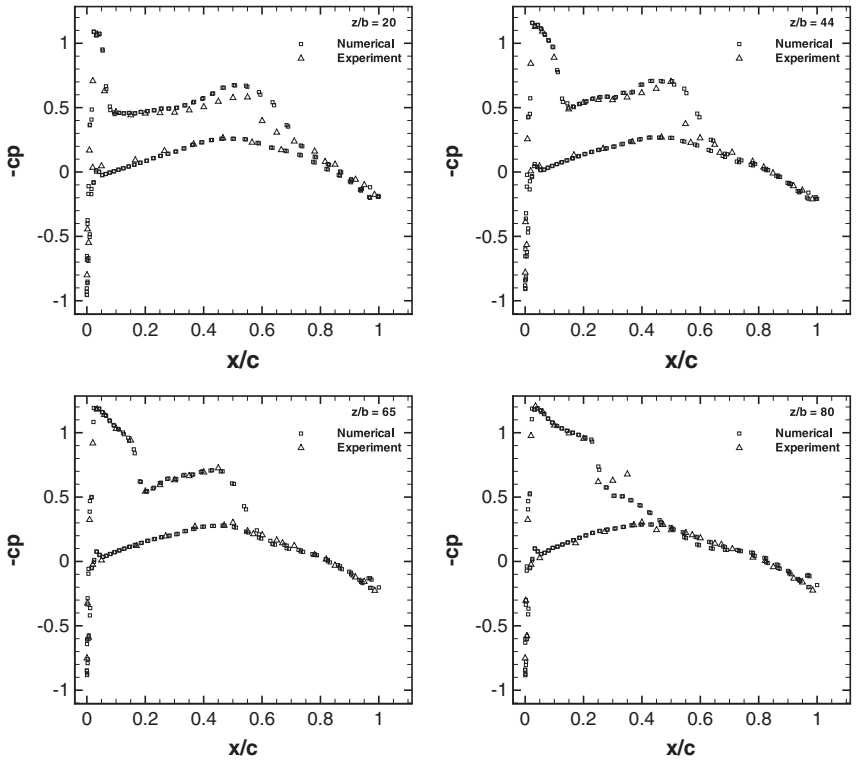


Figure 20. Cross section of the pressure coefficient at 20%, 44%, 65%, and 80% of the chord compared with the experimental results.

7. CONCLUSIONS AND PERSPECTIVES

In this paper, we have developed an algorithm that is able to deform linear $\mathbb{P}^1$ meshes onto valid quadratic ones. It assumes that the exact boundary surfaces are known (and are quadratic). This algorithm uses a linear elasticity analogy, but instead of using it with $\mathbb{P}^2$ Lagrange element, we apply it on a linear mesh with vertexes that are interpreted as the control points of Bézier (or NURBS) elements. This is much less expensive in terms of CPU cost, and also, in practice, the algorithm reveals to be very robust. No parameter is needed and the procedure is fully automatic.

The resulting curved mesh respects exactly the boundary. The generation of such curved meshes is automatic and without user-defined parameter. The quality of the elements is good enough to perform CFD simulations, including when boundary layers exist.

In all the examples we have considered so far, the surface description is of good quality. It is likely that some more work is needed when the surface description is not of enough quality or when the initial volume mesh is not of sufficient quality too. The extension to other type than simplexes seems feasible but this has to be evaluated, as well as the case of higher than quadratic elements.

APPENDIX A. HIGH-ORDER MESH FILE

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

- `Order` where k is the order of the mesh.
- `NumberOfP1Dofs ndof` with ndof the number of DOF of the associated P^1 mesh.

DOI: 10.1002/fld

- `Weights`
  `np`
  w_1
  `...`
  w_{np}
- `PkEdges`
- `PkTriangles`

ACKNOWLEDGEMENTS

R. Abgrall has been partially supported by the FP7 Advanced Grant # 226316 'ADDECCO' and by the Swiss FNFS grant #200021_153604/1. A. Froehly has been fully supported by the FP7 Advanced Grant # 226316 'ADDECCO'. The CFD numerical method and all the CFD simulations have been conducted by Dante de Santis (INRIA) who is warmly acknowledged. He has been funded by the FP7 STREP IDHIOM.

REFERENCES

1. Bassi F, Rebay S. High-order accurate discontinuous finite element solution of the 2D Euler equations. *Journal of Computational Physics* 1997; **138**(2):251–285.
2. Cockburn B, Shu C-W. The Runge–Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws. *RAIRO – Modélisation Mathématique et Analyse Numérique* 1991; **25**(3):337–361.
3. Cockburn B, Karniadakis GE, Shu C-W (eds.). *Discontinuous Galerkin Methods. Theory, Computation and Applications. 1st International Symposium on DGM, Newport, RI, USA, May 24–26, 1999*. Springer: Berlin, 2000.
4. Hughes TJR, Mallet M. A new finite element formulation for computational fluid dynamics. III: the generalized streamline operator for multidimensional advective- diffusive systems. *Computer Methods in Applied Mechanics and Engineering* 1986; **58**:305–328.
5. Hughes TJR, Mallet M, Mizukami A. A new finite element formulation for computational fluid dynamics. II. Beyond SUPG. *Computer Methods in Applied Mechanics and Engineering* 1986; **54**:341–355.
6. Hughes TJR, Cottrell JA, Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering* 2005; **194**(39–41):4135–4195.
7. Abgrall R, Larat A, Ricchiuto M. Construction of very high order residual distribution schemes for steady inviscid flow problems on hybrid unstructured meshes. *Journal of Computational Physics* 2011; **230**(11):4103–4136.
8. Vymazal M, Quintino T, Villedieu N, Deconinck H. High-order upwind residual distribution schemes on isoparametric curved elements. *Journal of Computational Physics* 2011; **230**(4):890–906.
9. Frey PJ, George P-L. *Mesh Generation. Application to Finite Elements* (2nd edn). ISTE: London; John Wiley: Hoboken, NJ, 2008.
10. Shephard MS, Flaherty JE, Jansen KE, Li X, Luo X, Chevaugeon N, Remacle J-F, Beall MW, O'Bara RM. Adaptive mesh generation for curved domains. *Applied Numerical Mathematics* 2005; **52**(2–3):251–271.
11. Sahni O, Luo XJ, Jansen KE, Shephard MS. Curved boundary layer meshing for adaptive viscous flow simulations. *Finite Elements in Analysis and Design* 2010; **46**(1–2):132–139. Mesh Generation - Applications and Adaptation.
12. Luo XJ, Shephard MS, O'bara RM, Nastasia R, Beall MW. Automatic p-version mesh generation for curved domains. *Engineering with Computers* 2004; **20**:273–285.
13. Dey S, O'Bara RM, Shephard MS. Towards curvilinear meshing in 3D: the case of quadratic simplices. *Computer-Aided Design* 2001; **33**(3):199–209.
14. Dey S, O'bara RM, Shephard MS. Curvilinear mesh generation in 3D. *Proceedings of the Eighth International Meshing Roundtable*, John Wiley & Sons, South Lake Tahoe, CA, 1999; 407–417.
15. George PL, Borouchaki H. Construction of tetrahedral meshes of degree two. *International Journal for Numerical Methods in Engineering* 2012; **90**(9):1156–1182.
16. Xie ZQ, Sevilla R, Hassan O, Morgan K. The generation of arbitrary order curved meshes for 3D finite element analysis. *Computational Mechanics* 2013; **51**:361–374.
17. Persson P-O, Peraire J. Curved mesh generation and mesh refinement using lagrangian solid mechanics. *Proceedings of the 47th AIAA Aerospace Sciences Meeting and Exhibit.* January 2009, AIAA paper, 209–949.
18. Remacle J-F, Toulorge T, Lambrechts J. Robust untangling of curvilinear meshes. In *Proceedings of the 21st International Meshing Roundtable*, Jiao X, Weill J-C (eds). Springer: Berlin Heidelberg, 2013; 71–83.
19. Abgrall R, De Santis D, Ricchiuto M. High order preserving residual distribution schemes for advection–diffusion scalar problems on arbitrary grids. *Rapport de recherche RR-8157*, INRIA, November 2012. submitted to *Siam Journal on Scientific Computing*.
20. Abgrall R, de Santis D, Ricchiuto M. High order preserving residual distribution schemes for the laminar and turbulent navier-stokes equations on arbitrary grids. *Journal of Computational Physics* 2013. submitted.
21. Abgrall R, Baurin G, Krust A, de Santis D, Ricchiuto M. Numerical approximation of parabolic problems by residual distribution schemes. *International Journal on Numerical Methods in Fluids* 2013; **71**(9):1191–1206.

DOI: 10.1002/fld

22. Abgrall R, Larat A, Ricchiuto M. Construction of very high order residual distribution schemes for steady inviscid flow problems on hybrid unstructured meshes. *Journal of Computational Physics* 2011; **230**(11):4103–4136.
23. Geuzaine C, Remacle J-F. Gmsh: a 3-D finite element mesh generator with built-in pre- and post-processing facilities, 2009.
24. Dobrzynski C, Frey P. Mmg3d: anisotropic tetrahedral remesher/moving mesh generation. (Available from: http://www.math.u-bordeaux1.fr/~cdobrzyn/logiciels/mmg3d.php) [Accessed on 2010].

DOI: 10.1002/fld