Automatic surface mesh generation for discrete models – A complete and automatic pipeline based on reparametrization

Pierre-Alexandre Beaufort [a,b,∗], Christophe Geuzaine [b], Jean-François Remacle [a]

[a] *Université catholique de Louvain, iMMC, Avenue Georges Lemaitre 4, 1348 Louvain-la-Neuve, Belgium*
[b] *Université de Liège, Montefiore Institute, Allée de la Découverte 10, B-4000 Liège, Belgium*

A B S T R A C T

Triangulations are an ubiquitous input for the finite element community. However, most raw triangulations obtained by imaging techniques are unsuitable as-is for finite element analysis. In this paper, we give a robust pipeline for handling those triangulations, based on the computation of a one-to-one parametrization for automatically selected patches of input triangles, which makes each patch amenable to remeshing by standard finite element meshing algorithms. Using only geometrical arguments, we prove that a discrete parametrization of a patch is one-to-one if (and only if) its image in the parameter space is such that all parametric triangles have a positive area. We then derive a non-standard linear discretization scheme based on mean value coordinates to compute such one-to-one parametrizations, and show that the scheme does *not* discretize a Laplacian on a structured mesh. The proposed pipeline is implemented in the open source mesh generator Gmsh, where the creation of suitable patches is based on triangulation topology and parametrization quality, combined with feature edge detection. Several examples illustrate the robustness of the resulting implementation.

© 2020 Elsevier Inc. All rights reserved.

1. Introduction

Engineering designs are often encapsulated in Computer Aided Design (CAD) systems. This is usually the case in automotive, shipbuilding or aerospace industries. The finite element method is the proeminent technique for performing analysis of these designs and this method requires a finite element mesh, i.e. a subdivision of CAD geometrical entities into a (large) collection of simple geometrical shapes such as triangles, quadrangles, tetrahedra and hexahedra, arranged in such a way that if two of them intersect, they do so along a face, an edge or a node, and never otherwise.

In CAD systems, the geometry of surfaces is described through a parametrization i.e. a mapping

$$\mathbf{x} \ : \ A \mapsto \mathbb{R}^3, \quad (u; v) \mapsto \mathbf{x}(u; v) \tag{1}$$

where $A \subset \mathbb{R}^2$ is usually a rectangular region $[u_0, u_1] \times [v_0, v_1]$. When finite element mesh generation procedures have access to such parametrizations $\mathbf{x}(u; v)$ of surfaces, it is in general a good idea to generate a planar mesh in the parametric domain A and map it in 3D. This way of generating surface meshes is called *indirect* [1], and is the predominant method for generating high-quality finite element surface meshes in a robust manner. This approach is in particular followed by the

* Corresponding author at: Université catholique de Louvain, iMMC, Avenue Georges Lemaitre 4, 1348 Louvain-la-Neuve, Belgium.
  E-mail address: pierre-alexandre.beaufort@uclouvain.be (P.-A. Beaufort).

open source mesh generator Gmsh [2], which directly interacts with CAD systems to get parametrizations $\mathbf{x}(u; v)$ as well as their derivatives (normals, curvatures...). The nature of the mappings $\mathbf{x}(u; v)$ that are provided by CAD systems is such that anisotropic planar meshing capabilities are required in order to be able to generate quality meshes in 3D. Those mappings may be very irregular and even singular, for example at the two poles of a sphere. Gmsh's surface planar mesh generators have been designed in such a way that they can handle very distorted metrics [3] while still providing high quality outputs.

In domains like geophysics or in bio-sciences, however, the geometry of the models is rarely described through CAD models. Most often, those geometries are produced through imaging (segmentation, voxelization, ...) whose geometrical output can be reduced to a triangulation. Several authors have proposed *direct* approaches [4,5] that modify this raw "geometrical" mesh to produce a "computational" mesh with elements of controlled shapes and sizes. The aim of this paper is to show that the *indirect* approach is also possible in this case, in which a (global) parametrization $\mathbf{x}(u; v)$ is not readily available. Starting from a triangulation, our aim is to build *a set of parametrizations* that form an atlas of the model, and which can be used as-is by existing finite element mesh generators.

This paper describes the complete pipeline that allows to build the atlas of the model together with the parametrizations of all its maps. It aims at being self-consistent, which makes it quite exhaustive. In §2, some theoretical background on mappings is presented. Then, §3 develops the concept of discrete parametrizations. A complete set of proofs based on purely geometrical arguments is given that assert the injectivity of the discrete maps that are used. The way Gmsh handles the input in order to ease the parametrization and meshing process is described within §4. We point out the drawback of a general processing of coarse discrete surfaces in §5, and discuss two ways to handle such coarse discretizations. Several examples are presented in Section §6, and conclusions are drawn in Section §7.

2. Mappings

A parametrization $\mathbf{x}(u; v)$ as defined in Equation (1) is regular if $\partial_u \mathbf{x}$ and $\partial_v \mathbf{x}$ exist and are linearly independent:

$$\partial_u \mathbf{x} \times \partial_v \mathbf{x} \neq \mathbf{0}$$

for any $(u; v) \in A$. In other words, $\mathbf{x}(u; v)$ is regular if and only if the Jacobian matrix

$$J = \frac{\partial \mathbf{x}}{\partial (u; v)} \in \mathbb{R}^{3 \times 2} \tag{2}$$

associated to $\mathbf{x}(u; v)$ has rank 2 $\forall (u; v) \in A$. The nature of the mapping $\mathbf{x}(u; v)$ is fully characterized by the *singular value decomposition* (SVD) of its Jacobian (2). Its singular values $\sigma_1 \geq \sigma_2 > 0$ allow to characterize $\mathbf{x}$:

- $\mathbf{x}$ is *isometric* if and only if $\sigma_1 = \sigma_2 = 1$,
- $\mathbf{x}$ is *conformal* if and only if $\frac{\sigma_2}{\sigma_1} = 1$,
- $\mathbf{x}$ is *equiareal* if and only if $\sigma_1 \sigma_2 = 1$.

Isometric parametrizations preserve essentially everything (lengths, areas and angles). With such nice properties, generating well shaped triangles in the planar domain A will lead to a well shaped mesh in 3D. Disappointingly, such length preserving mappings do not exist for surfaces that are not developable [6, Chapter 2, §4] i.e. that have non zero Gaussian curvatures.

Conformal mappings conserve angles. If $\mathbf{x}(u; v)$ is conformal, isotropy is preserved and standard isotropic mesh generators can do the surface meshing job. Again, the odds are against us: although it is possible to build conformal mappings for most surfaces, it is very difficult to ensure global injectivity of such mappings, even though conformal mappings are always locally injective. Thus, ensuring the global one-to-oneness of conformal mappings is still an open question [7].

Equiareal mappings have no interest in mesh generation. Thus, in general, mesh generators are faced with general parametrizations that do not preserve anything. This means that anisotropic planar mesh generators are required to generate well shaped meshes in 3D. Anisotropic mesh generators usually take as input a Riemannian metric field defined in each $(u; v)$ of A. If the aim is to produce an isotropic 3D mesh with a mesh size defined by an isotropic mesh size field $h(\mathbf{x}(u; v))$, the metric tensor that is used by the mesh generator will be

$$M(u; v) = \frac{J^T J}{h^2}.$$

Let us assume for example that the surface to be meshed is an ellipsoid. Fig. 1 shows a 3D surface mesh that is adapted to the maximal curvature of the surface as well as its counterpart in the parameter plane of the ellipsoid. The particular ellipsoid of Fig. 1 is $e = 7$ times wider in the x direction than in the two other directions y and z. Its parametrization (which is standard to most CAD systems) is

$$x(u, v) = e \sin u \, \sin v$$

$$y(u, v) = \sin u \, \cos v$$

$$z(u, v) = \cos u$$

a. Mesh in the parameter plane A.

b. Mesh in $\mathbb{R}^3$.

c. Largest singular value σ_1 in A.

d. Smallest singular value σ_2 in A.

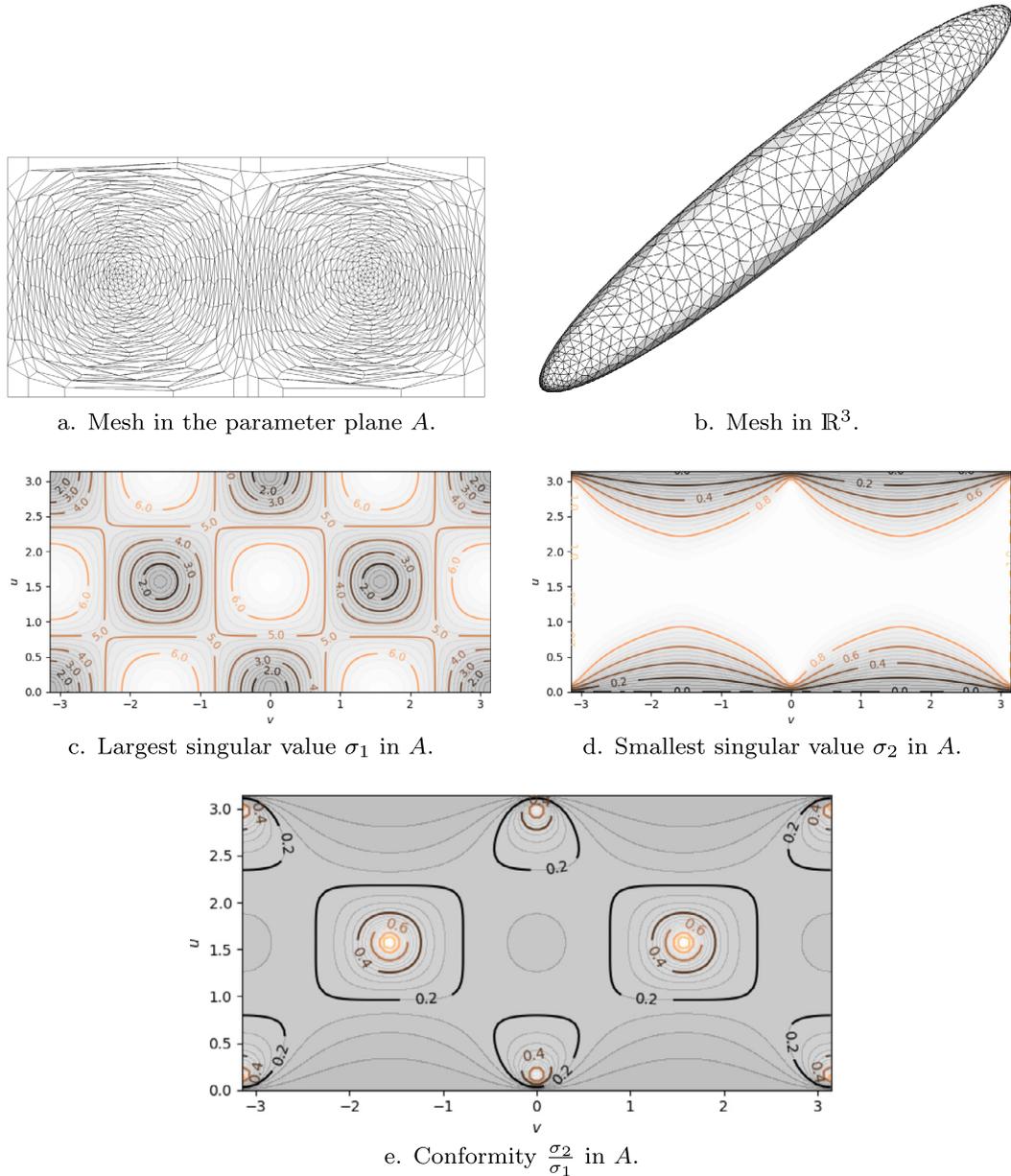e. Conformity $\frac{\sigma_2}{\sigma_1}$ in A.

Fig. 1. The case of an ellipsoid. Figure a shows the mesh of the ellipsoid in the parameter space while Figure b shows the same mesh in the 3D space. Figures c and d show the largest and smallest singular values σ_1 and σ_2 of the jacobian J. Figure e shows the non conformity parameter σ_2/σ_1.

where $u \in [0, \pi]$ is the inclination and $v \in [-\pi, \pi[$ is the azimuth. The metric tensor associated to that mapping is

$$M = \frac{1}{h^2} \begin{pmatrix} \cos^2 v (e^2 \sin^2 v + \cos^2 v) + \sin^2 u & \sin u \sin v \cos u \cos v (e^2 - 1) \\ \sin u \sin v \cos u \cos v (e^2 - 1) & \sin^2 u (e^2 \cos^2 v + \sin^2 v) \end{pmatrix}. \tag{3}$$

The mapping is obviously not regular when $u = 0$ and when $u = \pi$. This is surprisingly not so much of a problem for mesh generators. In [3], authors propose a way to slightly modify meshing procedures in order to deal with singular mappings such as the one of the ellipsoid. The metric field (3) is anisotropic (see Fig. 1e) and non-uniform. Yet it is smooth and smoothness of mappings is the most important feature of $\mathbf{x}(u; v)$ in order to allow mesh generators to do a good job. When generating a mesh in an indirect fashion, a planar mesh, possibly anisotropic, is generated in the parameter plane A. Then, one may think that this planar mesh is mapped in 3D through $\mathbf{x}(u; v)$, which is not true: only corners of the triangles are mapped in 3D and those corners are connected together with 3D straight lines that are not the actual mapping of 2D straight lines. In the best case scenario, any 2D straight line connecting points $(u_a; v_a)$ and $(u_b; v_b)$ corresponds the geodesic between those two points. When the metric M is locally constant, geodesics are straight lines and the indirect

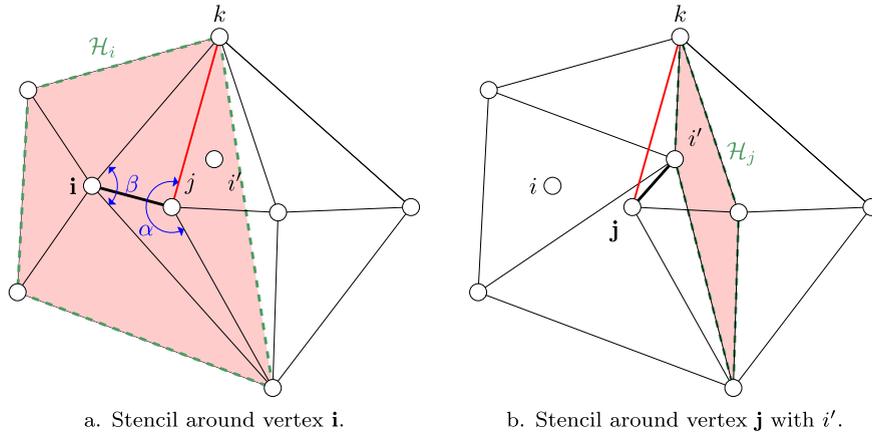a. Stencil around vertex **i**.   b. Stencil around vertex **j** with i'.

Fig. 2. Sketch of proof for monotonicity.

meshing approach gives good results. When the metric varies rapidly along one given edge, then indirect meshing becomes difficult. In CAD systems, parametrizations are always smooth and indirect mesh generation is always possible.

3. Discrete parametrizations

In Section §2, we have shown that having a smooth parametrization was the condition to allow indirect surface meshing. CAD systems provide smooth parametrizations but CAD models are not the only geometrical representations that are available in engineering analysis. In many domains of engineering interest, geometries of models are described by triangulations. We call such models *discrete models*.

Assume a triangulation T with $\#p$ nodes (vertices), $\#e$ edges and $\#t$ triangles. Finding a parametrization of T consists in assigning to every vertex p_i of the triangulation a pair of coordinates $(u_i; v_i)$. If every triangle (p_i, p_j, p_k), with $p_\bullet \in \mathbb{R}^3$[1] of the triangulation has a positive area in the $(u; v)$ plane, then the parametrization is injective.

A parametrization of T onto a subset of $A \subset \mathbb{R}^2$ exists if the triangulation corresponds to the one of a planar mesh. Assume that triangulation T is simply connected with $\#b$ boundaries and $\#h$ vertices on those boundaries. Then the surface is parameterizable if and only if

$$\#t = 2(\#p - 1) + 2(\#b - 1) - \#h.$$

In what follows, we present some existing material that is detailed in numerous publications [8–10]. The main interest of this section is that we take here the point of view of the numerical geometer. The main result about the one-to-oneness of mappings is proven without using one single theorem of analysis such as maximum principles of Radó-Kneser-Choquet theorem [11].

Consider an internal vertex i of T and $J(i)$ the set of indices whose the corresponding nodes are connected to the node i (in other words, edge (i, j) exists $\forall j \in J(i)$). The value of the parametric coordinates (u_i, v_i) at vertex i will be computed as a weighted average of the coordinates (u_j, v_j) of its neighboring vertices:

$$\sum_{j \in J(i)} \lambda_{ij}(u_i - u_j) = 0 \ , \quad \sum_{j \in J(i)} \lambda_{ij}(v_i - v_j) = 0 \tag{4}$$

where λ_{ij} are coefficients. This scheme is a called a *difference scheme* that involves only differences $(u_i - u_j)$, with $j \in J(i)$. If every λ_{ij} is positive, values of u_i and v_i are convex combinations of their surrounding values. In a geometrical point of view, it actually means that point (u_i, v_i) lies in the convex hull $\mathcal{H}_i$ of its neighboring vertices.

With that assumption, it is easy to prove that the mapping provided by any positive scheme of the type (4) is one-to-one. Let us consider a triangle (i, j, k) in the parameter plane $(u; v)$, Fig. 2. If edge (j, k) belongs to $\mathcal{H}_i$, that triangle (i, j, k) is obviously positive.

On the other hand, if (j, k) is inside $\mathcal{H}_i$, as it is the case in Fig. 2, then (j, k) does not belong to $\mathcal{H}_i$ and moving i to i' creates an inverted triangle (i', j, k) while keeping $\mathcal{H}_i = \mathcal{H}_{i'}$. In this case, i' is inside $\mathcal{H}_i$ while triangle (i', j, k) is inverted. It is easy to see that moving i to i' implies that j would be outside $\mathcal{H}_j$ which is in contradiction with the hypothesis that each vertex is inside its convex hull. Vertex j being inside $\mathcal{H}_i$ implies that $\alpha > \pi$. The sum of the four angles of a quadrangle is 2π. This implies that $\beta < \pi$ which implies that edge (i, k) belongs to $\mathcal{H}_j$. So, moving i to i' puts j outside $\mathcal{H}_j$.

---

[1] In what follows, a triangle is denoted by the indices of its nodes, i.e. (i, j, k) instead of (p_i, p_j, p_k).
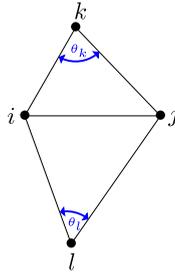
Fig. 3. Definitions of θ_k and θ_l for the difference scheme corresponding to the linear Galerkin approach.

Now see what happens on the outer boundary ∂A of the $(u; v)$ domain A. There, points have no neighboring hull. Yet, assuming that ∂A is convex, then all vertices of ∂A that are connected to internal vertices belong their convex hull and no internal vertex cannot be situated outside A. This last part of the proof has some similarities with the one of [12].

This means that a positive scheme applied to a convex domain implies that the discrete parametrization is one-to-one. In our case, we will always choose ∂A as the unit circle.

Now, the right choice of the λ_{ij} is of outmost importance for ensuring a good parametrization. Our use of parametrization is meshing. The first and non negociable property of the discrete parametrization is one-to-oneness. We thus choose a positive scheme and a $(u; v)$ domain that is a unit circle. The second priority is smoothness, we will develop that aspect below. The icing on the cake would be conformity but, as noted in §2, Gmsh's mesh generators are comfortable with anisotropic mappings and we will not put any effort on that aspect of the game (our aim is not texture mapping like in computer graphics, so we are OK to map squares on circles), even though some sort of conformity will come naturally.

3.1. Parametrization smoothness

We look here for a smooth function $\mathbf{x}(u, v)$ i.e. a continuous function which derivatives are smooth as well because we want σ_1 and σ_2 to be smooth and σ_1 and σ_2 are by-products of the metric i.e. a tensor computed using the first derivatives of $\mathbf{x}(u, v)$. Tutte's barycentric mapping [9] consist in choosing $\lambda_{ij} = 1$. This choice leads to very irregular mappings that are useless for mesh generation purposes. The idea that has been advocated by many authors [7,13] is to solve a partial differential equation which solutions are inherently smooth. For example, the solution of Laplace equations on domains with smooth boundaries and with smooth boundary conditions are C^∞ and it is indeed a good idea to choose the λ_{ij} in such a way the difference operator (4) is a discrete version of the Laplace operator.

3.2. Laplace smoothing using $\mathcal{P}^1$ finite elements

The standard P^1 finite element formulation of the Laplace problem is well known for more than a half of a century. In the early days, some authors [14] have written coefficients $\lambda_{ij}^{\text{FEM}}$ in a quite geometrical fashion (see Fig. 3):

$$\lambda_{ij}^{\text{FEM}} := \frac{1}{2} \left(\frac{\cos(\theta_k)}{\sin(\theta_k)} + \frac{\cos(\theta_l)}{\sin(\theta_l)} \right). \tag{5}$$

For sake of completeness, the so called "cotangent formula" (5) is fully derived in Appendix A. Coefficients $\lambda_{ij}^{\text{FEM}}$ of (5) may be negative for $\theta_\bullet \in \left(\frac{\pi}{2}; \pi \right)$, which could lead to scheme that is not provably injective. This is the very old result that states that the maximum principle satisfied by solutions of Laplace equations is only guaranteed *a priori* by finite elements computed on acute triangulations, i.e. triangulations without obtuse angles. Acute triangulations are a sufficient condition for injectivity. Yet it is not necessary and it is indeed complicated to find examples where finite elements fail to provide one-to-one parametrizations. Disappointingly, in the world of mesh generation, limit cases that happen once in a thousand have to be avoided. So, we will not use finite elements for parametrizing our surfaces.

3.3. Mean value coordinates

A continuous function f is a solution of Laplace equation $\nabla^2 f = 0$ on an open set $A \subset \mathbb{R}^2$ if and only if, for every $\mathbf{x} \in A$, $f(\mathbf{x})$ is equal to the average value of f over every circle of radius r $\mathcal{C}(\mathbf{x}; r)$ that fully belongs to A:

$$f(\mathbf{x}) = \frac{1}{2\pi r} \int\limits_{\mathcal{C}(\mathbf{x}; r)} f(\mathbf{x}') \, d\mathbf{x}'. \tag{6}$$

This principle states that the extrema of the mapping are located on the boundary of the domain, and that there is not local extremum inside the domain.
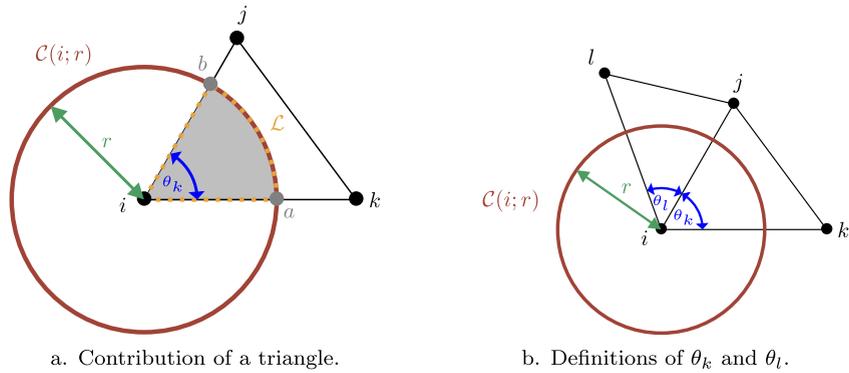
a. Contribution of a triangle.   b. Definitions of θ_k and θ_l.

Fig. 4. Derivation of the difference scheme corresponding to the mean value coordinates.

In [12], Floater proposes a way to compute λ_{ij} that actually mimics property (6): this scheme is called mean value coordinates. In this paper, we re-derive Floater's λ_{ij} corresponding to mean value coordinates using a finite element point of view. According to (6), the value f_i is the average of values $f(\mathbf{x})$ along a circle $\mathcal{C}(i; r)$ of radius r centered on i (see Fig. 4). A linear interpolation $f(x; y) = \sum_j f_j \phi_j(x; y)$ is assumed over each triangle $\mathcal{T}_{ijk}$. We are going to compute the contribution of triangle $\mathcal{T}_{ijk}$ for (6)

$$\theta_k r\, f_i = \int_{\widehat{ab}} f_i \phi_i + f_j \phi_j + f_k \phi_k \; ds$$

where θ_k is the angle between edges $[ij]$ and $[ik]$, and $\widehat{ab}$ is the circle arc of $\partial\mathcal{C}(i, r)$ contains in $\mathcal{T}_{ijk}$, Fig. 4a. Since $\phi_i + \phi_j + \phi_k = 1$,

$$\underbrace{\left(\theta_k r - \int_{\widehat{ab}} \phi_i \; ds\right)}_{\int_{\widehat{ab}} \phi_j + \phi_k \, ds} f_i - \int_{\widehat{ab}} \phi_j \; ds f_j - \int_{\widehat{ab}} \phi_k \; ds f_k = 0$$

which gives

$$\underbrace{\int_{\widehat{ab}} \phi_j \; ds}_{\lambda_{ij}} (f_i - f_j) + \underbrace{\int_{\widehat{ab}} \phi_k \; ds}_{\lambda_{ik}} (f_i - f_k) = 0$$

over $\mathcal{T}_{ijk}$.

Linear shape function ϕ_j associated to node j in $\mathcal{T}_{ijk}$ corresponds to

$$\phi_j(x; y) = \frac{y}{y_j}$$

where y is the vertical coordinate relative to edge $[ik]$ and y_j is the y-coordinate of node j. We compute the integral of y over $\widehat{ab}$ from the contour $\mathcal{C}$ composed of $\widehat{ab}$, $[bi]$ and $[ia]$

$$\int_{\mathcal{C}} y \; ds$$

From normal vector of $\mathcal{C}(i; r)$ $\hat{n} = \frac{1}{r}(x; y)$, we get

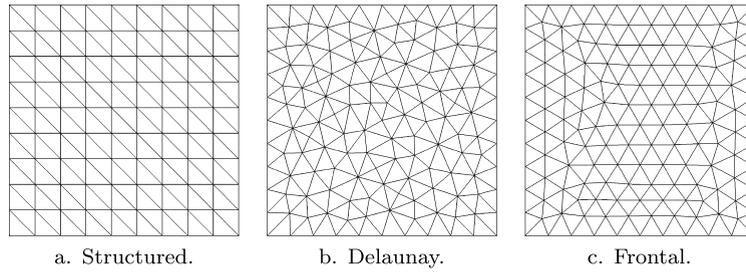$$\int_{\mathcal{L}} \frac{y}{r} \; ds = \int_{\mathcal{L}} \hat{n} \cdot \mathbf{e_y} \; ds$$

a. Structured.   b. Delaunay.   c. Frontal.

Fig. 5. Types of meshes on a square.

with $\mathbf{e_y} = (0; 1)$. The divergence of $\mathbf{e_y}$ is obviously zero, and owing to the divergence theorem

$$\int_{\mathcal{L}} \frac{y}{r}\, ds = \int_{\mathcal{R}(\mathcal{L})} \nabla \cdot \mathbf{e_y}\, dx\, dy = 0$$

where $\mathcal{R}(\mathcal{L})$ is the region surrounded by $\mathcal{L}$ (gray area, Fig. 4a). The integral along the circle arc $\overset{\frown}{ab}$ is then equal to the opposite of integrals along edges $[bi]$ and $[ia]$ of triangles $\mathcal{T}_{ijk}$

$$\int_{\overset{\frown}{ab}} \underbrace{\hat{n} \cdot \mathbf{e_y}}_{\frac{y}{r}}\, ds = -\left(\int_{[ia]} \underbrace{\hat{n} \cdot \mathbf{e_y}}_{-1}\, ds + \int_{[bi]} \underbrace{\hat{n} \cdot \mathbf{e_y}}_{\cos(\theta_k)}\, ds \right)$$

$$= -(-r + r\cos(\theta_k))$$
$$= r(\cos(\theta_k) - 1)$$

Since $y_j = l_{ij}\sin(\theta_k)$, with l_{ik} the length of edge $[ij]$

$$\int_{\overset{\frown}{ab}} \phi_k\, ds = r^2 \frac{\tan\left(\frac{\theta_k}{2}\right)}{l_{ij}}$$

Choosing a radius r small enough (i.e. smaller than the smallest edge within the triangulation) allows to simplify the finite scheme (5) by r^2, which means that the scheme does not depend on the circle of integration. The coefficient λ_{ij} is then given by

$$\lambda_{ij} = \frac{\tan\left(\frac{\theta_k}{2}\right) + \tan\left(\frac{\theta_l}{2}\right)}{l_{ij}} \tag{7}$$

We notice that $\lambda_{ij} > 0$, $\forall \theta_\bullet \in (0; \pi)$. The difference scheme (4) with (7) builds linear injective mappings. This monotone scheme is not symmetric, except on equilateral triangulations.

At that point, one can raise the question of the actual accuracy of the MVC scheme for discretizing Laplace equation, which is our guarantee of smoothness. A convergence experiment[2] has been performed on a square $[0; 1] \times [0; 1]$ on various meshes (Fig. 5) using the standard technique of manufactured solutions. We choose $f(x; y) = \sin(2\pi x)\cosh(2\pi y)$ whose laplacian $\nabla^2 f$ is zero.

Fig. 6 shows that MVC scheme does not exhibit the usual FEM convergence. The absence of symmetry of the MVC scheme implies that only $\mathcal{O}(h)$ convergence is observed for the L^2 norm. Yet, the MVC scheme seems to converge on all meshes except the structured one. This behavior is due to pollution.

3.4. Boundary conditions

We consider 3D surfaces that are topologically equivalent to a disk with $\#b - 1$ internal boundaries. The parametric domain that is considered is always a unit disk

$$A = \left\{ (u; v) \in \Re^2 : u^2 + v^2 < 1 \right\}.$$

The setup is described in Fig. 7. Dirichlet boundary conditions are applied on $\mathbf{x}(\partial A)$ that actually ensure that the u, v

---

[2] The experiment has been performed with the Gmsh API, given in supplementary material.
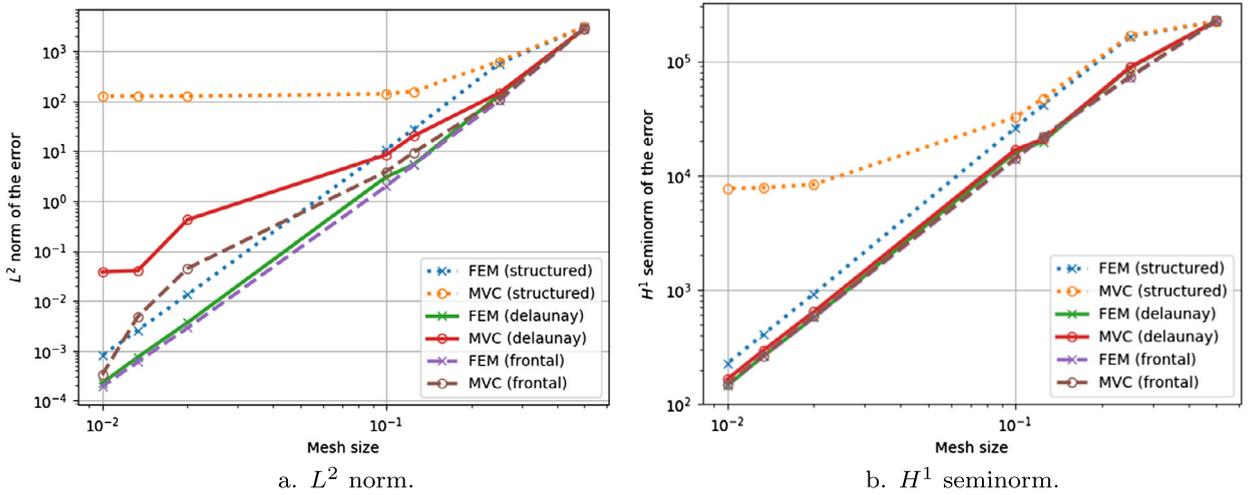
a. L^2 norm.
b. H^1 seminorm.

Fig. 6. h-convergence of discrete schemes (4) with (5) VS (7) on mesh types of Fig. 5.
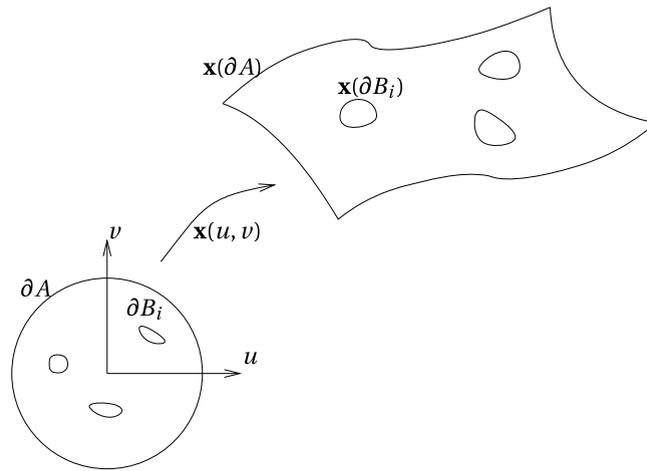

Fig. 7. A 3D domain that is topologically equivalent to a disk with 3 internal boundaries and its parametric domain A.

coordinates on $\mathbf{x}(\partial A)$ correspond to the unit circle

$$\delta A = \left\{ (u; v) \in \Re^2 : u^2 + v^2 = 1 \right\}.$$

We should now decide on what boundary conditions to apply on the other boundaries δB_i. The issue here is that we do not know *a priori* their position in the parameter plane. We could decide their position and insert #$b - 1$ small circles inside A. Yet, this would lead to a parametrization that is quite distorted. Another option is to apply the smoother as is to every internal points, including the ones on the internal boundaries. This indeed corresponds to imposing homogeneous Neumann boundary conditions on every internal boundary. It is indeed easy to prove that this choice still leads to a one-to-one parametrization. One first thing to note is that if every ∂B_i is convex and if we use a convex combination map like (7), then the mapping is one-to-one.

Assume that points $p_1, p_2, \ldots, p_k$ form a closed loop in the parameter plane and that every point lies in the convex hull of its neighbors, such as Fig. 8. Then, polygon $(p_1, p_2, \ldots, p_k)$ is convex. Indeed, if every three consecutive points i, j, k of such a loop form an angle α_j that is greater or equal to π, then the edges $(i; j)$ and $(j; k)$ lay in the convex hull $\mathcal{H}_j$. If it is true for every point of the loop corresponding to the hole, then its loop in the parameter plane is convex. From §3, we know that a positive scheme produce a one-to-one parametrization. Hence, if no condition are imposed on the holes - which corresponds to Neumann condition within FEM formulation - the parametric representation of those holes correspond to convex loop, whatever the initial shape of holes (i.e. even if they were concave).

Fig. 9b shows a concave domain with a concave hole that is mapped using (7) and where homogeneous Neuman boundary conditions were applied to the internal boundary. In this case, $\partial_n u = \partial_n v = 0$ on the internal boundary and the parametrization is close to be singular because the two tangent vectors are nearly parallel: both of them are *weakly* orthogonal to the boundary (see Fig. 9b)!
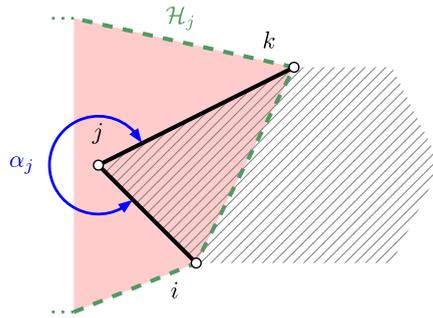
Fig. 8. Three consecutive points belonging to a loop describing an hole (hatched area) in *A*.

Another option consists of *filling the holes*, which leads to better results in practice (see Fig. 9b).

A heuristic to fill holes is to link each vertex lying on the hole to a *pseudo center* **c** of the hole. This pseudo center corresponds to the center of the circle associated to the hole, Fig. 10. The hole is approximated by a circle which circumference $2\pi r$ corresponds to the perimeter of the hole $\sum_j l_j$. The vertices defining the hole are then assumed to lie on such a circle. New triangles are then defined, by connecting those vertices to the pseudo center of the hole. The angle α_j defined by $\angle \mathbf{v_i c v_{i+1}}$ is assumed to be equal to $\frac{l_j}{r}$. Since the triangles filling the hole are assumed to share **c**, they are isosceles. All those assumptions enable to average the parametric coordinates of vertices lying on the hole, such that there was no hole. The triangles filling the hole are not explicitly built.

The heuristic performs well, even if the hole is concave and badly shaped, Fig. 9b. The improvement compared to the homogeneous Neumann condition is obvious, Fig. 9a. Actually, some parametric triangles of Fig. 9a are too tight for meshing purposes.

The drawback of filling holes is that it increases the connectivity of the linear system enabling the computation of the underlined parametrization. Indeed, the pseudo centers corresponds to extra unknowns which are related to the corresponding unknowns along each hole. Hence, the corresponding rows within the matrix representing the linear system may have a lot of nonzero. In order to avoid a memory overflow, a threshold of the potential connectivity is set: if there are too many vertices on a hole, Neumann boundary conditions are set. Otherwise, this hole is filled with the pseudo center.

4. Gmsh's pipeline for discrete surface meshing

The specifications of Gmsh's algorithm for the generation of meshes on discrete surfaces are the following

- A conforming "watertight" geometrical triangulation is given as input.
- A mesh with user specified mesh size parameters is given as output by Gmsh where all mesh vertices lies exactly on the input triangulation.

In Gmsh's new pipeline, the problem of surface meshing is divided in two stages: (i) a pre-processing stage and (ii) a mesh generation stage. In order to explain the usefulness the two stages of the pipeline, a relatively simple example will serve as a common theme to illustrate the various treatments that have to be undergo by a rough geometric triangulation to become a high quality finite element mesh.

Fig. 11 shows the geometric triangulation of a "Batman" object that is connected to a torus.

In Gmsh's pipeling, a rough geometrical triangulation is taken as input. A triangulation like the one of Fig. 11 cannot be processed as is for a number of reasons.

4.1. Detecting feature edges

The geometrical triangulation of Fig. 11 is composed of a list of triangles, period. The first part of our pre-processing is to detect feature edges of the geometry that should be present in the final mesh. We use here a simple angle criterion to detect feature edges. After detecting feature edges, a first version of the final atlas is created. Fig. 12 shows the Batman geometry where feature edges have been created for all edges that have two adjacent triangles with normals separated by an angle of more than 40 degrees. A first version of the final topology of the domain is created with model faces that are bounded by the feature edges. After the computation of feature edges, curvature tensors are computed at every vertex of every surface using [15].

4.2. Creating the atlas

At that point, we are not yet ready to compute the atlas of the model i.e. the final boundary representation of our model together with the parametrization of all its model surfaces. As explained in §3 every model surface of the atlas should have
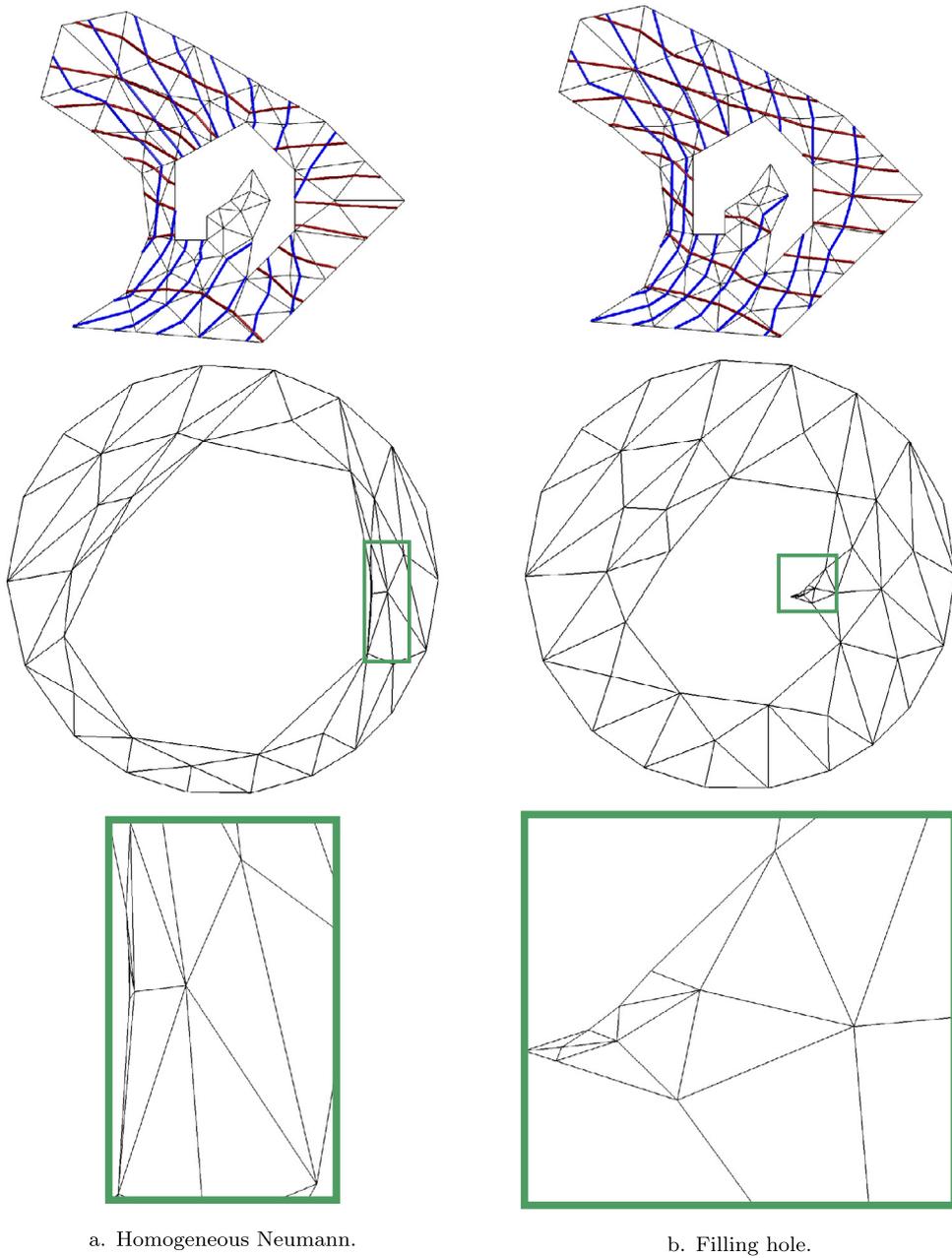
a. Homogeneous Neumann.   b. Filling hole.

Fig. 9. Demonstration of filling a concave hole with the circle assumption. Top: parametrization over the (discrete) geometry (u: red isolines, v: blue isolines). Bottom: triangles within the computed parametric space. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)
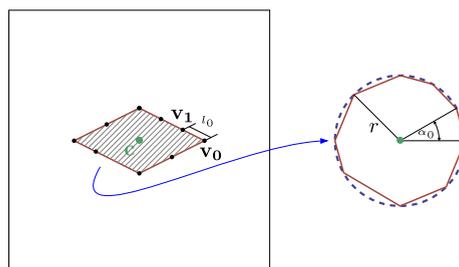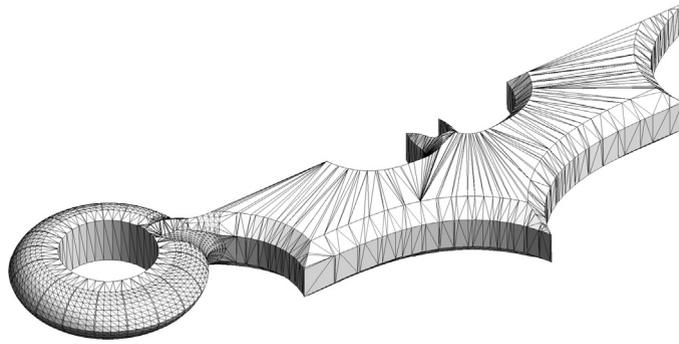

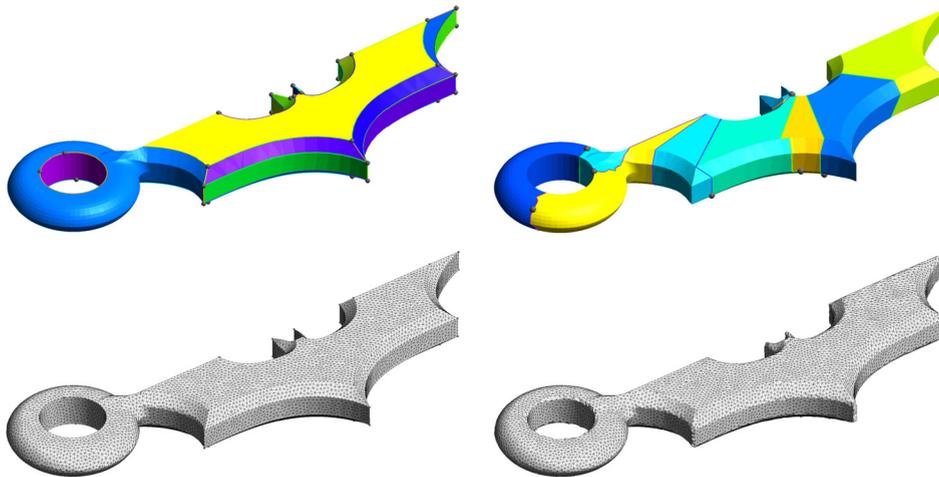**Fig. 10.** Exampled filling hole (hatched area).

Fig. 11. The Batman geometry.

Fig. 12. Top left Figure shows the final model with feature edges detection (threshold angle of 40 degrees). Bottom left Figure shows a uniform mesh on that model. Right Figures show the final model and mesh without feature edges detection. The domain has been split automatically in such a way that every model face has the right topology.

the right topology. In this following step, we ensure that every map of the atlas has this right topology. When a surface has a larger genus, it is split in two parts using a graph partitioning technique [16]. This operation is applied up to the point when every surface is parametrizable.

It is also known that surfaces with large aspect ratios may lead to parametrizations that have non distinguishable coordinates. When the parametrization is computed, we also ensure that parametric triangles are not too small i.e. that their area is not close to machine precision [13]. If it is the case, the surface is split in two.

For large models, we also split surfaces that contain a too large number of triangles (typically 100,000). Computing mean value coordinates require to solve a non symmetrical system of equations and one of the design goals of the parametrization process is to be fast.

Fig. 12 (top right) shows the decomposition that has been done on the Batman model without pre-computing feature edges.

4.2.1. The final BREP

At that point, the input triangulation has been transformed into a proper boundary representation that has a valid topology and for which each face has been parametrized. All those topological and geometrical information are now saved in the version 4 of the output mesh format of Gmsh. This "extended" mesh file can be used as input to Gmsh's surface mesh generators. Fig. 12 (bottom images) show meshes for both models generated using feature edges and automatic splitting.

5. Improving parametrization on coarse discrete surfaces

The methodology that has been presented before is general and applies to triangulated surfaces of arbitrary complexity. Yet, geometrical triangulations of CAD surfaces may not be sufficiently dense to allow a smooth parametrization. For example, a good geometrical triangulation of a cylinder may not contain internal vertices as depicted on Fig. 13. We have computed the parametrization of this cylinder using mean value coordinates and the result is presented in Fig. 13a. Even if
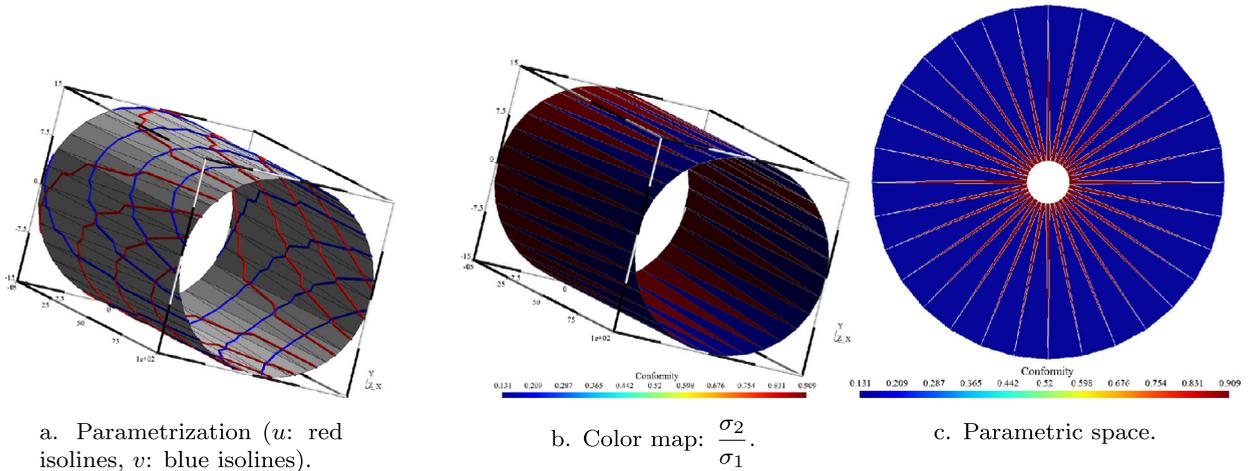
a. Parametrization (u: red isolines, v: blue isolines).

b. Color map: $\dfrac{\sigma_2}{\sigma_1}$.

c. Parametric space.

Fig. 13. Parametrization on coarse stl triangulations: a cylinder.

a. Parametrization (u: red isolines, v: blue isolines).

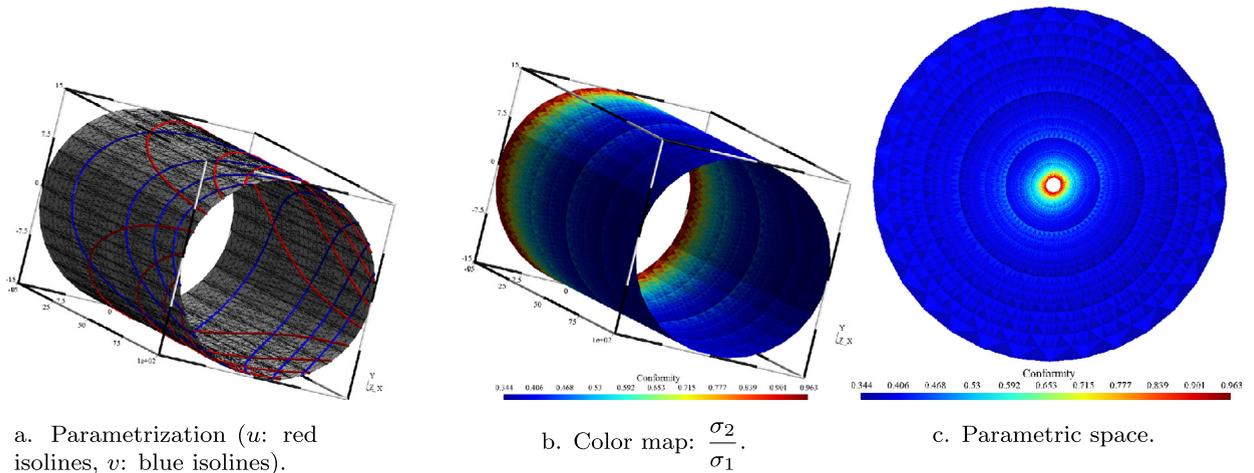b. Color map: $\dfrac{\sigma_2}{\sigma_1}$.

c. Parametric space.

Fig. 14. Parametrization on refined stl triangulations: the cylinder (5 iterations).

this parametrization is said "moderately noised", it cannot be used for mesh generation purposes. Figs. 13b and 13c show conformity indicator $\frac{\sigma_2}{\sigma_1}$ both on the real and parameter space of the cylinder.

From this observation, a numerical analyst would suggest two ways to improve the computation: refining the solution (i.e. the input mesh), either increasing the order of the approximation (i.e. second order).

5.1. Refinement by longest edge bisection

We refine the geometrical triangulation without changing its geometry i.e. only using edge splits. We use here a variant of the well known longest edge bisection process [17]: edges to be split are tagged and the longest edge of the list is split, then the second longest edge is split and the process continues until the shortest edge of the list is split. We repeat the process several times up to the point all inner edges respect a length threshold. Fig. 14a shows the new geometrical mesh of the cylinder.

In order to illustrate the effect of this refinement on the parametrization, we have pre-computed a "good mesh" of the cylinder in the 3D space (see Figure Fig. 15a). This good mesh has been inverse-mapped onto the parameter spaces of the non refined cylinder and on the refined cylinder. While doing that, we can see the meshes that should have been created by Gmsh's surface meshers in both parameter planes to obtain the same "good mesh". Fig. 15b shows the mesh in the parameter plane of the non refined geometrical cylinder: it contains series of elongated triangles followed by isotropic ones, illustrating the too great variability of the conformity parameter. In Fig. 15c, the mesh is anosotropic but element shapes are locally uniform and any good anisotropic mesher is able to generate that kind of mesh.
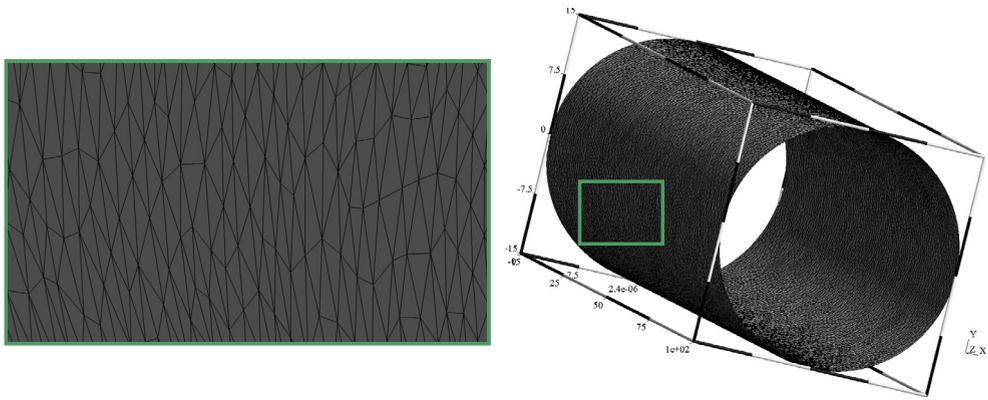
a. Good mesh on cylinder.
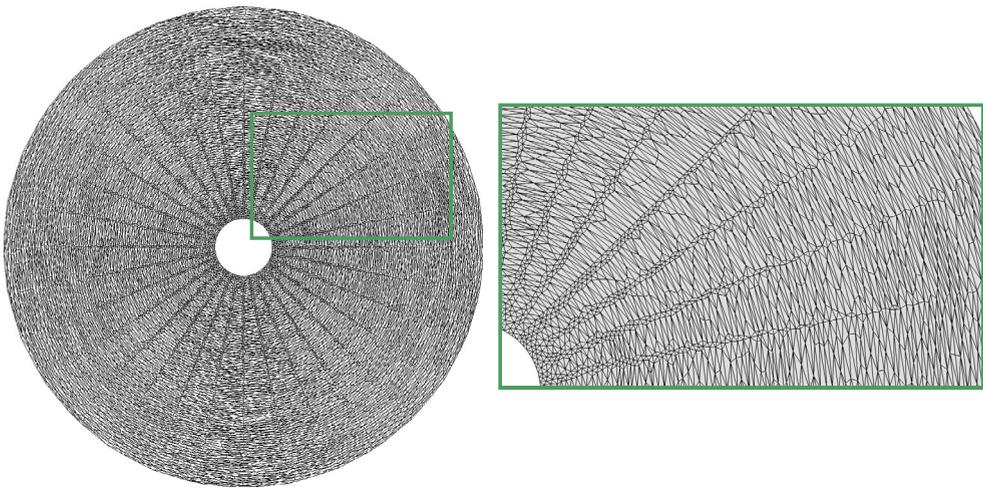
b. Mapping within the parametric space, *without* preprocessing (cf. Fig. 13c).
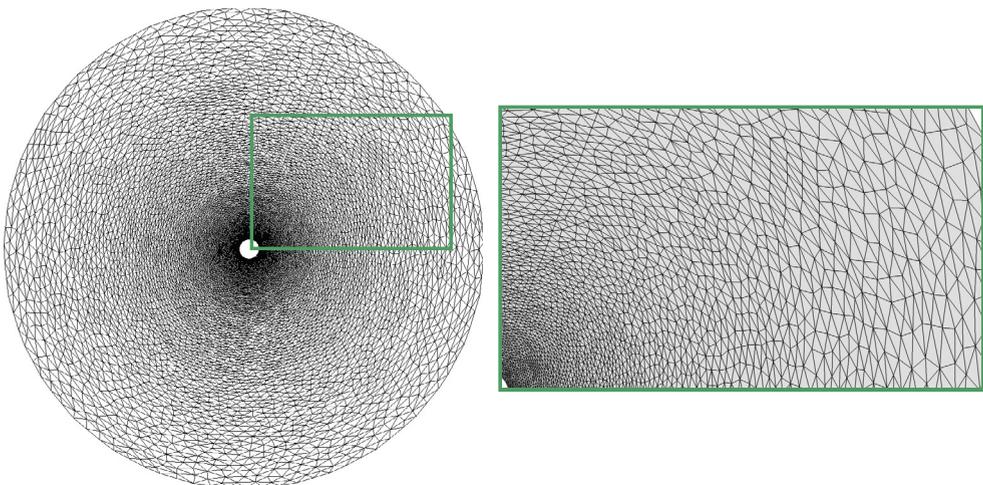
c. Mapping within the parametric space, *with* preprocessing (cf. Fig. 14c).

Fig. 15. Effect on mapping a good mesh on a parametrization with (b) and without (c) edge refinement preprocessing.
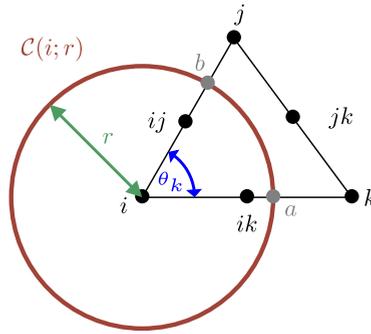
Fig. 16. Sketch for quadratic approximation of $\lambda_{ij}^{\mathcal{P}^2}$.

5.2. Second order approximation

As in the piecewise linear approximation (see §3.3), we derive λ_{ij} from Lagrange $\mathcal{P}^2$ function shapes

$$\theta_k r\, f_i = \int_{\widehat{ab}} f_i\phi_i + f_j\phi_j + f_k\phi_k + f_{ij}\phi_{ij} + f_{jk}\phi_{jk} + f_{ik}\phi_{ik}\, ds$$

where $\phi_\bullet$ are the Lagrange $\mathcal{P}^2$ finite element shape functions, which defined with the barycentric coordinates (v_i, v_j, v_k) [18, Chapter 1,§1.2.4]

$$\begin{cases} \phi_a = v_a(2v_a - 1),\ a \in \{i, j, k\} \\ \phi_{ab} = 4v_a v_b,\ a, b \in \{i, j, k\} : a \neq b \end{cases}$$

Assigning coordinates relative to v_i, Fig. 16

$$\begin{aligned} v_i &= (0; 0) \\ v_j &= (l_{ij}\cos(\theta_k); l_{ij}\sin(\theta_k)) \\ v_k &= (l_{ik}; 0) \end{aligned}$$

Again, $\phi_i + \phi_j + \phi_k + \phi_{ij} + \phi_{jk} + \phi_{ik} = 1$ enables us to write

$$\underbrace{\left(\theta_k r - \underbrace{\int_{\widehat{ab}} \phi_i\, ds}_{\widehat{ab}}\right)}_{\int_{\widehat{ab}} \phi_j + \phi_k + \phi_{ij} + \phi_{jk} + \phi_{ik}} f_i - \int_{\widehat{ab}} \phi_j\, ds f_j - \int_{\widehat{ab}} \phi_k\, ds f_k$$

$$- \int_{\widehat{ab}} \phi_{ij}\, ds f_{ij} - \int_{\widehat{ab}} \phi_{jk}\, ds f_{jk} - \int_{\widehat{ab}} \phi_{ik}\, ds f_{ik} = 0$$

which gives

$$\underbrace{\int_{\widehat{ab}} \phi_j\, ds(f_i - f_j)}_{\lambda_{ij}} + \underbrace{\int_{\widehat{ab}} \phi_k\, ds(f_i - f_k)}_{\lambda_{ik}}$$

$$+ \underbrace{\int_{\widehat{ab}} \phi_{ij}\, ds(f_i - f_{ij})}_{\lambda_{i(ij)}} + \underbrace{\int_{\widehat{ab}} \phi_{jk}\, ds(f_i - f_{jk})}_{\lambda_{i(jk)}} + \underbrace{\int_{\widehat{ab}} \phi_{ik}\, ds(f_i - f_{ik})}_{\lambda_{i(ik)}} = 0$$

a. Kuratowski graph of type I.

b. Graph depicting Lagrange $\mathcal{P}^2$ dof's on a triangle.

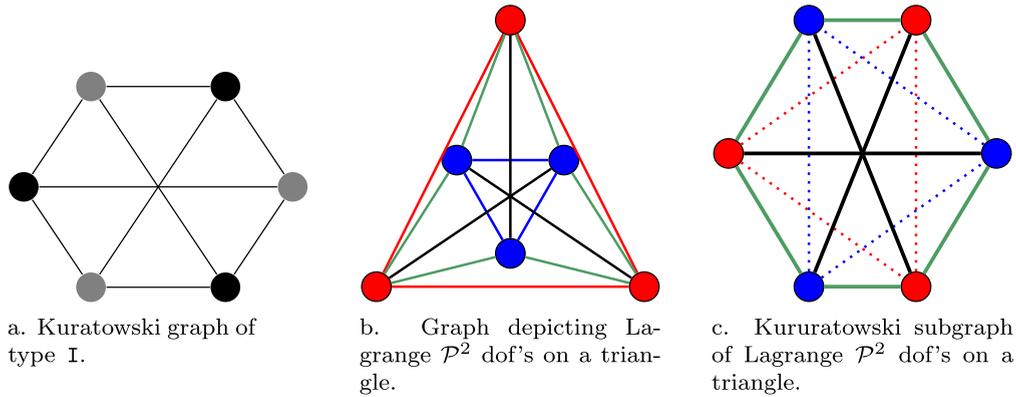c. Kururatowski subgraph of Lagrange $\mathcal{P}^2$ dof's on a triangle.

Fig. 17. Graph corresponding to Lagrange $\mathcal{P}^2$ dof's on a triangle has no planar representation.
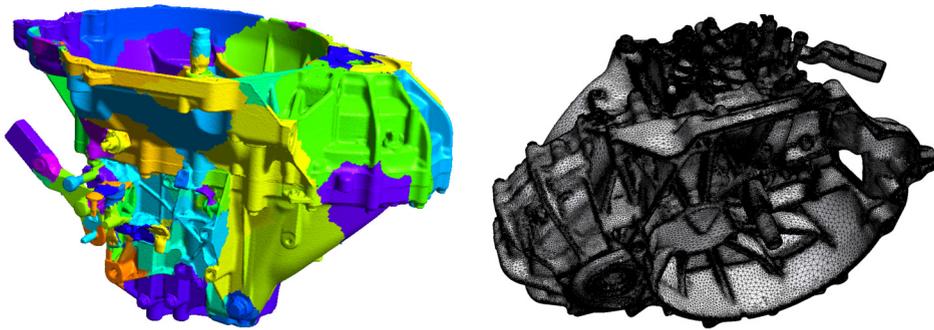

Fig. 18. Complex scanned mechanical part. The initial triangulation (left) that contains $797,666$ triangles has been split into 194 surfaces that are parametrizable. The mesh on the right that contains $1,762,388$ triangles and has been adapted to the curvature of the original discrete surface. It has been generated by Gmsh in 640 seconds, including IO's.

We use SymPy[19] to compute $\lambda_{ij}^{\mathcal{P}^2}$ (code in supplementary material)

$$\lambda_{ij}^{\mathcal{P}^2} = \frac{r^2}{l_{ij}^2 \, \sin^2(\theta_k)} \left((l_{ij} - r) \cos(\theta_k) \sin(\theta_k) + r\theta_k - l_{ij} \sin(\theta_k) \right) \tag{8}$$

We should derive the other coefficients $\lambda_{\bullet}^{\mathcal{P}^2}$, but something is wrong with (8). We cannot get rid of r within the expression. It means that the coefficients give the average for a certain circle of radius r. Yet, it has to be for *any* circle, whatever the radius. It is then not possible to derive $\lambda^{\mathcal{P}^2}$ for a monotone scheme.

Actually, graph theory states such a result. Lagrange $\mathcal{P}^2$ degrees of freedom on a triangle may be depicted by a 3-connected graph, Fig. 17b. Tutte [9, §4] claims that any graph having a Kuratowski subgraph is nonplanar. Fig. 17a corresponds to a Kuratowski graph. A graph is planar if it can be drawn on a plane, in such a way that its edge intersects only on vertices of the graph. It means that each vertex of the graph may correspond to a convex combination of its neighbors, which we aim. However, Fig. 17b has such a Kuratowski subgraph, Fig. 17c. The graph of Fig. 17b has no planar representation. Hence, it means it is not possible to write Lagrange $\mathcal{P}^2$ scheme which is monotone.

6. Examples

In this section, several complex examples are presented that show the level of robustness that has been attained by our methodology. (See Figs. 18–21.) The examples that have been chosen in order to challenge our algorithm and push it to the limit.

7. Conclusion

This paper has demonstrated the Gmsh's ability to remesh robustly poor quality triangulations, for the purpose to run finite element analysis. Gmsh's pipeline essentially relies on the one-to-oneness of parametrization, where conformity is not mandatory since a mesher has to deal with anisotropic meshes. We have shown that such a discrete parametrization is possible only if the corresponding mapping orients all parametric triangles in the same way.
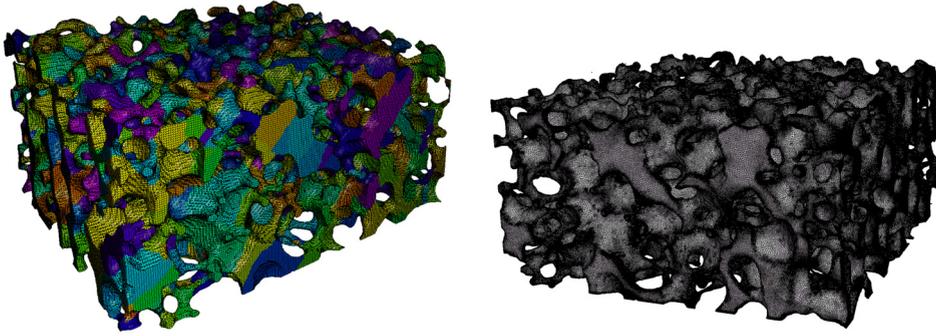
Fig. 19. X-ray tomography image of a silicon carbide foam (from P. Duru, F. Muller and L. Selle, IMFT, ERC Advanced Grant SCIROCCO). The initial triangulation (left) that contains $1,288,116$ triangles has been split into $1,802$ surfaces that are parametrizable. The mesh on the right contains $4,922,322$ triangles and has been adapted to the curvature of the original discrete surface. It been generated by Gmsh in $1,187$ seconds, including IO's.
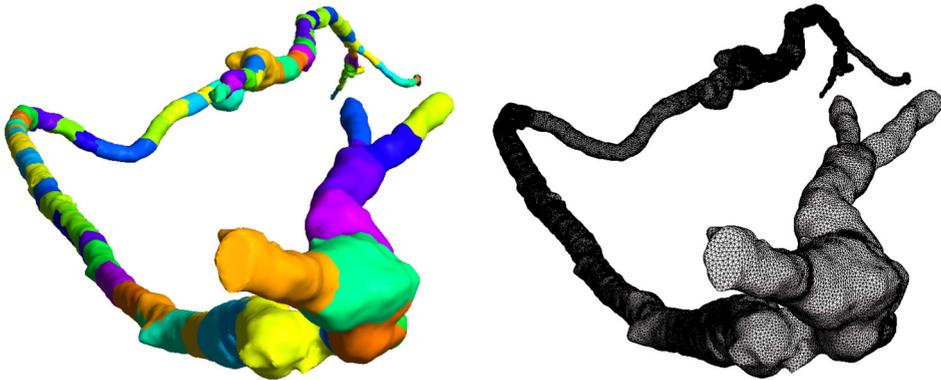

Fig. 20. CT scan of an artery. The initial triangulation (left) that contains $63,468$ triangles has been split into 101 surfaces that are parametrizable. Most of the cuts were done because of the large aspect ratio of the tubular domains. The uniform mesh on the right that contains $170,692$ triangles has been generated by Gmsh in 22 seconds, including IO's.
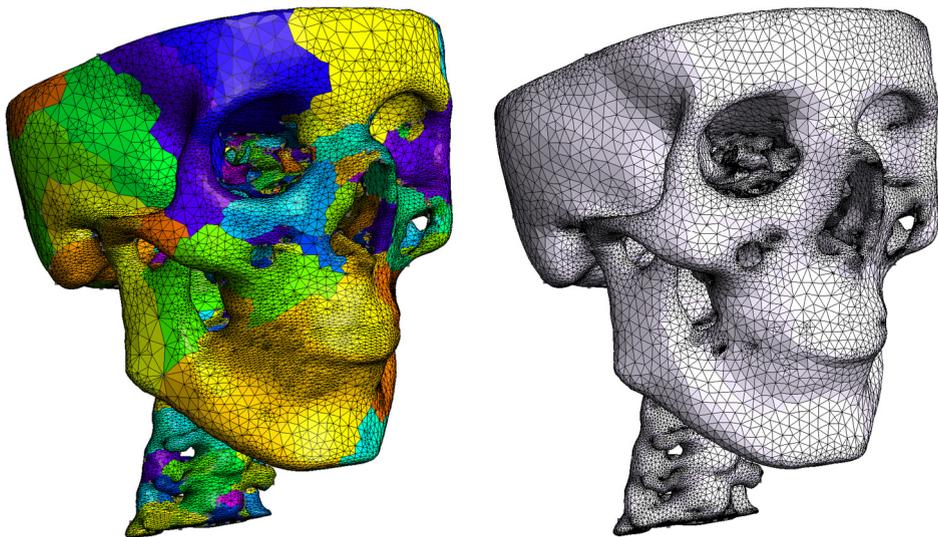

Fig. 21. Remeshing of a skull. The initial triangulation (left) that contains $142,742$ triangles has been split into 715 surfaces that are parametrizable. The mesh on the right is adapted to the surface curvature and contains $323,988$ triangles and has been generated by Gmsh in 58 seconds, including IO's.

Based on the mean value theorem and assuming a linear approximation, we have derived the well-known mean value coordinates. We performed a convergence test of the corresponding scheme: it does not discretize properly a Laplacian on a structured mesh; otherwise, it has the expected convergence for a scheme that is not symmetric. We have proved that if Neumann conditions are set along the boundary of holes (within a triangulation), the mean value coordinates give parametric holes which boundary is convex. Since it unnecessarily deforms the parametrization, we gave an heuristic that fills the holes as they were circular in order to produce better parametrizations.

With one simple but graphic example, we have shown the effect of feature edge detection on the atlas creation. We have discussed how to improve the parametrization of a coarse triangulation: the only way is to perform a longest edge bisection before parametrization. We have shown there is no Lagrange $\mathcal{P}^2$ version of the mean value coordinates. Finally, several difficult examples are exhibited as a demonstration of the robustness of Gmsh's pipeline.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

Appendix A. Derivation of the FEM scheme for harmonic mapping

Continuous harmonic maps minimize the *Dirichlet energy*

$$\int_{P_i} |\nabla \phi_i|^2 d\mathbf{x}$$

of the parametrization ϕ_i on the patch P_i. In other words, it minimizes the distortion between the patches and their planar representation.

It is possible to write a Laplacian as a finite difference scheme

$$\nabla^2 f|_i \approx \sum_{j}^{\#i} \lambda_{ij} (f_i - f_j)$$

It is a linear approximation of a Laplace operator at a vertex i. Indeed, the Laplace operator corresponds to the Euler-Lagrange equations derived from the Dirichlet energy

$$\int |\nabla f|^2 d\mathbf{x} \approx \int ||\sum_{j} f_j \nabla \phi_j||^2 d\mathbf{x}$$

$$\frac{d}{df} \int |\nabla f|^2 d\mathbf{x}\bigg|_i \approx 2\int \sum_{j} f_j \nabla \phi_j \cdot \nabla \phi_i \, d\mathbf{x}$$

with $\phi_{\bullet}$ denoting the linear function shape associated to a node. On a triangle $\mathcal{T}_{ijk}$, knowing that $\phi_i + \phi_j + \phi_k = 1$ over $\mathcal{T}_{ijk}$

$$\int_{\mathcal{T}_{ijk}} f_i \nabla \phi_i \cdot \nabla \underbrace{(1 - \phi_j - \phi_k)}_{\phi_i} + f_j \nabla \phi_i \cdot \nabla \phi_j + f_k \nabla \phi_i \cdot \nabla \phi_k \, d\mathbf{x}$$

Rewriting last relation with terms $(f_i - f_j)$ and $(f_i - f_k)$, we obtain

$$\lambda_{ij} = -\int_{\mathcal{T}_{ijk}} \nabla \phi_i \cdot \nabla \phi_j \, d\mathbf{x}$$

which corresponds to the standard Galerkin finite element.

On triangle $\mathcal{T}_{ijk}$ (Fig. 3)

$$\nabla \phi_i \cdot \nabla \phi_j = |\nabla \phi_i||\nabla \phi_j| \, \cos(\pi - \theta_k)$$
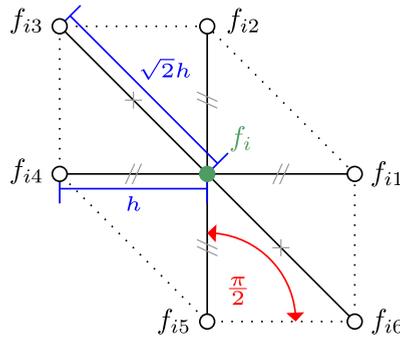
where

Fig. 22. Stencil within a structured mesh.

$$|\nabla\phi_i| = \frac{1}{l_{ik}\sin(\theta_k)} \qquad\qquad |\nabla\phi_j| = \frac{1}{l_{jk}\sin(\theta_k)}$$

with $l_{\bullet k}$ the length of edge $[\bullet k]$. Knowing that $|\mathcal{T}_{ijk}| = \frac{1}{2}l_{ik}l_{jk}\sin(\theta_k)$

$$-\int_{\mathcal{T}_{ijk}} \nabla\phi_i \cdot \nabla\phi_j \, d\mathbf{x} = \frac{1}{2}\frac{\cos(\theta_k)}{\sin(\theta_k)}$$

Adding the contribution of $\mathcal{T}_{ilj}$

$$\lambda_{ij}^{\text{FEM}} := \frac{1}{2}\left(\frac{\cos(\theta_k)}{\sin(\theta_k)} + \frac{\cos(\theta_l)}{\sin(\theta_l)}\right).$$

Appendix B. MVC difference scheme on a structured mesh is not a Laplacian

The MVC difference scheme relative to f_i corresponds to

$$\frac{\sqrt{2}}{h}\left(4f_i - (f_{i1} + f_{i2} + f_{i4} + f_{i5})\right) + \frac{2-\sqrt{2}}{h}\left(2f_i - (f_{i3} + f_{i6})\right) = 0 \qquad (B.1)$$

The first term of (B.1) (without the coefficient) corresponds to the well known linear combination of a centered finite difference to approximate a Laplacian. However, the second term does not approximate a continuous Laplacian. Indeed, the Taylor expansion of f_{i3} and f_{f_i6} is

$$f_{i3} = f_i - \left.\frac{\partial f}{\partial x}\right|_i h + \left.\frac{\partial f}{\partial y}\right|_i h + \left.\frac{\partial^2 f}{\partial x^2}\right|_i h^2 + \left.\frac{\partial^2 f}{\partial y^2}\right|_i h^2 - \left.\frac{\partial^2 f}{\partial x\partial y}\right|_i h^2 + \text{hot}$$

$$f_{i6} = f_i + \left.\frac{\partial f}{\partial x}\right|_i h - \left.\frac{\partial f}{\partial y}\right|_i h + \left.\frac{\partial^2 f}{\partial x^2}\right|_i h^2 + \left.\frac{\partial^2 f}{\partial y^2}\right|_i h^2 - \left.\frac{\partial^2 f}{\partial x\partial y}\right|_i h^2 + \text{hot}$$

Hence,

$$f_{i3} + f_{i6} - 2f_i = \overbrace{2\left.\frac{\partial^2 f}{\partial x^2}\right|_i h^2 + 2\left.\frac{\partial^2 f}{\partial y^2}\right|_i h^2}^{2h^2\,\nabla^2 f|_i} - 2\left.\frac{\partial^2 f}{\partial x\partial y}\right|_i h^2$$

Because of that last term, the MVC scheme on a structured mesh such as Fig. 22 is not the approximation of a continuous Laplacian. (See Fig. 23.)

Appendix C. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jcp.2020.109575.

References

[1] H. Borouchaki, P. Laug, P.-L. George, Parametric surface meshing using a combined advancing-front generalized Delaunay approach, Int. J. Numer. Methods Eng. 49 (1–2) (2000) 233–259.

[2] C. Geuzaine, J.-F. Remacle, Gmsh: a three-dimensional finite element mesh generator with built-in pre-and post-processing facilities, Int. J. Numer. Methods Eng. 79 (11) (2009) 1309–1331.
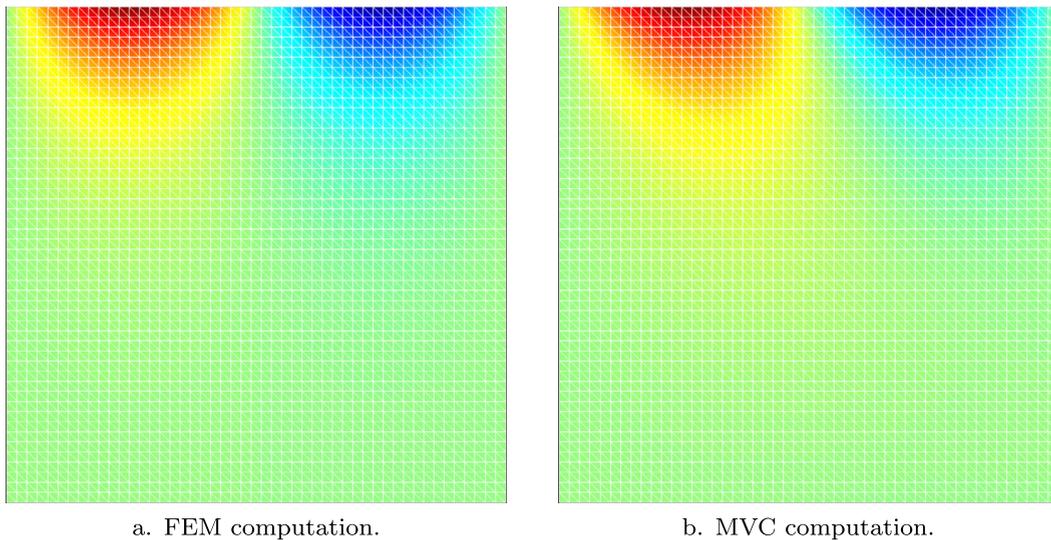
a. FEM computation. b. MVC computation.

Fig. 23. Approximations of $\nabla^2 f = 0$ on a structured mesh over a square $[0; 1] \times [0; 1]$.

[3] J. Remacle, C. Geuzaine, Gmsh's approach to robust mesh generation of surfaces with irregular parameterizations, in: 28th Internation Meshing Roundatable, Sandia National Laboratories, October 15-17, 2019.

[4] P. Frey, Yams a Fully Automatic Adaptive Isotropic Surface Remeshing Procedure, 2001.

[5] E. Béchet, J.-C. Cuilliere, F. Trochu, Generation of a finite element mesh from stereolithography (stl) files, Comput. Aided Des. 34 (1) (2002) 1–17.

[6] D.J. Struik, Lectures on Classical Differential Geometry, Courier Corporation, 1961.

[7] B. Lévy, S. Petitjean, N. Ray, J. Maillot, Least Squares Conformal Maps for Automatic Texture Atlas Generation, ACM Transactions on Graphics (TOG), vol. 21, ACM, 2002, pp. 362–371.

[8] M.S. Floater, K. Hormann, Surface parameterization: a tutorial and survey, in: Advances in Multiresolution for Geometric Modelling, Springer, 2005, pp. 157–186.

[9] W.T. Tutte, How to draw a graph, Proc. Lond. Math. Soc. 3 (1) (1963) 743–767.

[10] J.-F. Remacle, C. Geuzaine, G. Compere, E. Marchandise, High-quality surface remeshing using harmonic maps, Int. J. Numer. Methods Eng. 83 (4) (2010) 403–425.

[11] P. Dirac, The Lorentz transformation and absolute time, Physica 19 (1—12) (1953) 888–896, https://doi.org/10.1016/S0031-8914(53)80099-6.

[12] M.S. Floater, Mean value coordinates, Comput. Aided Geom. Des. 20 (1) (2003) 19–27.

[13] E. Marchandise, C.C. de Wiart, W. Vos, C. Geuzaine, J.-F. Remacle, High-quality surface remeshing using harmonic maps - part II: surfaces with high genus and of large aspect ratio, Int. J. Numer. Methods Eng. 86 (11) (2011) 1303–1321.

[14] R.J. Duffin, Distributed and lumped networks, J. Math. Mech. (1959) 793–826.

[15] S. Rusinkiewicz, Estimating curvatures and their derivatives on triangle meshes, in: Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004, 3DPVT 2004, IEEE, 2004, pp. 486–493.

[16] G. Karypis, V. Kumar, MeTis: unstructured graph partitioning and sparse matrix ordering system, version 4.0, http://www.cs.umn.edu/~metis, 2009.

[17] M.-C. Rivara, New longest-edge algorithms for the refinement and/or improvement of unstructured triangulations, Int. J. Numer. Methods Eng. 40 (18) (1997) 3313–3324.

[18] A. Ern, J.-L. Guermond, Theory and Practice of Finite Elements, vol. 159, Springer Science & Business Media, 2013.

[19] A. Meurer, C.P. Smith, M. Paprocki, O. Čertík, S.B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J.K. Moore, S. Singh, et al., Sympy: symbolic computing in python, PeerJ Comput. Sci. 3 (2017) e103.