Annexes au rapport technique

Contents

Appendix A

Gmsh

ULg-ACE and UCL-MMC provide the ONELAB consortium with Gmsh, an open-source three-dimensional finite element grid generator with a build-in CAD engine and post-processor. The development of Gmsh began in 1996; below we summarize the general design of Gmsh and of its modules. We describe in some detail the Application Programming Interface (API) that will be used as the basis for the abstract CAD, Meshing and post-processing interface of the ONELAB project.

Gmsh is built around four modules: geometry, mesh, solver and post-processing. Each module can be controlled either interactively using the GUI or using the built-in scripting language. The design of all four modules relies on a simple philosophy—be *fast*, *light* and *user-friendly*.

Fast: on a standard personal computer at any given point in time Gmsh should launch instantaneously, be able to generate a "larger than average" mesh (compared to the standards of the finite element community; say, one million tetrahedra in 2011) in less than a minute, and be able to visualize such a mesh together with associated post-processing datasets at interactive speeds.

Light: the memory footprint of the application should be minimal and the source code should be small enough so that a single developer can understand it. Installing or running the software should not depend on any non-widely available third-party software package.

User-friendly: the graphical user interface should be designed in such a way that a new user can create simple meshes in a matter of minutes. In addition, the code should be robust, portable, scriptable, extensible and thoroughly documented—all features contributing to a user-friendly experience.

The technical choices made to achieve these sometimes conflicting design objectives are detailed in [11]. Of particular note is that Gmsh is entirely written in standard C++ (both the kernel and the user interface, which is based on FLTK and OpenGL), that the kernel uses BLAS (through the GNU Scientific Library) for most of the basic linear algebra robust geometrical predicates in critical portions of the algorithms.

Gmsh is released under the GNU General Public License and is part of several official Linux distributions (most notably Debian). It can be built either as a stand-alone program or as a library on most computer architectures and operating systems, from Windows laptops to Macintosh workstations to large HPC Linux clusters.

Appendix B

Abstract CAD/Meshing Interface

Gmsh never had the ambition of becoming a solid modeling platform that competes with the few well-established, state of the art CAD engines like Parasolid or CATIA. The native Gmsh CAD engine thus has only a limited set of features, well suited for dealing with simple academic geometries. Yet, over the years, Gmsh has been used by an increasing number of people in industry, and a strong demand emerged from this user community for Gmsh to be able to mesh industrial CAD models.

One option for addressing this demand is to use exchange files, such as IGES (Initial Graphics Exchange Specification), VRML (Virtual Reality Markup Language) or STEP (STandard for the Exchange of Product model data). However, the use of such exchange file formats has always been a cause of trouble in engineering design offices, mainly because internal data structures of CAD systems are usually much richer than those in the exchange formats. The necessary simplifications of geometries as well as the importance of modeler tolerances that are not taken into account in exchange files lead to the time-consuming need to "fix" most of these exchange files before any meshing can be performed.

In Gmsh, we thus chose to deal with CAD engines differently, by providing *native* access to the underlying CAD models—without translation files (a similar approach is used in CAPRI [12]). For that, the geometry module is based on a set of abstract data structures that enables us to represent the topology of *any* solid model. Gmsh can then be extended to use new CAD engines simply by deriving these abstract data structures for each new engine.

B.1 Topological Entities

Any 3-D model can be defined using its Boundary Representation (BRep): a volume (called *region*) is bounded by a set of surfaces, and a surface is bounded by a series of curves; a curve is bounded by two end points. Therefore, four kinds of *model entities* are defined:

1. Model Vertices G_i^0 that are topological entities of dimension 0,
2. Model Edges G_i^1 that are topological entities of dimension 1,
3. Model Faces G_i^2 that are topological entities of dimension 2,
4. Model Regions G_i^3 that are topological entities of dimension 3.

Model entities are topological entities, i.e., they only deal with adjacencies in the model, and we use a bi-directional data structure for representing the graph of adjacencies. In this representation, a model entity G_i^d of dimension d holds one lists of upward adjacencies $G_j^{d+1}(G_i^d)$, i.e., all its adjacent entities of dimension $d+1$, and one list of downward adjacencies of dimension $d-1$, $G_j^{d-1}(G_i^d)$. Schematically, we have

$$G_i^0 \rightleftharpoons G_i^1 \rightleftharpoons G_i^2 \rightleftharpoons G_i^3.$$

This representation is said to be complete because any model entity is able to build its list of adjacencies of any dimension using local operations, i.e., without having to do a complete traversal of the adjacency graph of the model.

B.2 Geometrical Description

Each model entity G_i^d has a shape, a geometry. More precisely, it is a manifold of dimension d that is embedded in 3-D space. (Note that the overall geometric model may itself be non-manifold: Gmsh supports non-manifold features such as embedded curves and surfaces and connected volumes. Some non-manifold examples will be shown in the mesh generation Section B.4.)

The geometry of a model entity depends on the solid modeler for its underlying representation. Solid modelers usually provide a parametrization of the shapes, i.e., a mapping $\vec{x} \in R^d \mapsto \vec{p} \in R^3$:

1. The geometry of a model vertex G_i^0 is simply its 3-D location $\vec{x}_i = (x_i, y_i, z_i)$.
2. The geometry of a model edge G_i^1 is its underlying curve $\mathcal{C}_i$ with its parametrization $\vec{p}(t) \in \mathcal{C}_i$, $t \in [t_1, t_2]$.
3. The geometry of a model face G_i^2 is its underlying surface $\mathcal{S}_i$ with its parametrization $\vec{p}(u, v) \in \mathcal{S}_i$. Note that, for any curve $\mathcal{C}_j$ that is on a surface $\mathcal{S}_i$, mesh generation procedures require the ability to reparametrize any point $\vec{p}(t) \in \mathcal{C}_j$ on the surface $\mathcal{S}_i$, i.e., to compute the mapping $u = u(t)$ and $v = v(t)$. Gmsh either uses a brute force algorithm to compute the direct mapping $x = x(t)$, $y = y(t)$ and $z = z(t)$ and its inverse $u = u(x, y, z)$ and $v = v(x, y, z)$ (see Figure B.1), or, when the underlying CAD system provides it, the direct reparametrization of a point on a model face (i.e., a function that directly computes $u = u(t)$ and $v = v(t)$).
4. The geometry associated to a model region is R^3.

Solid modelers usually provide an API for the creation, manipulation, interrogation and storage of 3-D models. To perform mesh generation only a small subset of this API has to be interfaced—only some of the interrogation functions are necessary. In order to get the full functionality of Gmsh, only five CAD-system dependent interrogation functions have to be implemented for the model edge (see Figure B.2). For example, it is mandatory to be able to evaluate the mapping $\vec{p}(t) \in \mathcal{C}$ on the curve as well as the tangent vector $\vec{t}(t) = \partial_t \vec{p}(t)$. For the model face, only four functions have to be overloaded in order to enable 2-D mesh generation (see Figure B.3). Note that the default 2-D algorithm does not require the computation of derivatives of the surface parametrization, so that the function `GFace::tangent` is not

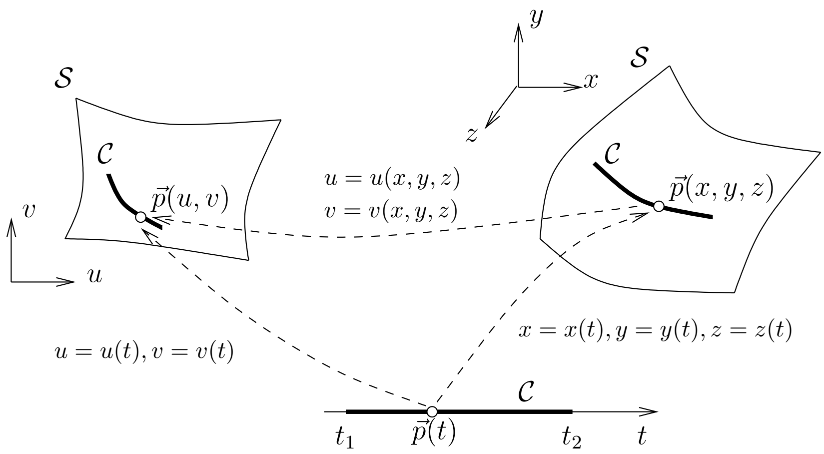


Figure B.1: Point $\vec{p}$ located on the curve $\mathcal{C}$ that is itself embedded in surface $\mathcal{S}$.

strictly required (a description of this 2-D algorithm is presented in Section B.7). The other 2-D algorithms available in Gmsh, as well as most of the available 2-D meshers (e.g. `bamg` [13] or `blsurf` [14]), make use of derivatives of the parametrization.

B.3 Solid Model

A Gmsh model is simply built as a list of model entities, each one of which possibly of different underlying geometries. In fact, as mentioned before, several CAD models can co-exist in the same Gmsh model. Mixing parts coming from different CAD engines can be very interesting in practice: for example, complex, non parametrizable parts designed with one CAD modeler (say, a full airplane model) can be extended with a parametrizable part written using the scripting language of the Gmsh native modeler (say, a radar antenna whose design one wishes to optimize). The overall model can then be discretized and optimized without any CAD translation.

B.4 Mesh Generation in Gmsh

For the description of the mesh generation process, let us consider the CAD model of a propeller presented in Figure B.4. The model has been created with the OpenCascade solid modeler and has been loaded in Gmsh in its native format (brep). The model contains 101 model vertices, 170 model edges, 76 model faces and one model region.

B.5 Mesh Size Field and Quality Measures

Let us define the mesh size field $\delta(x, y, z)$ as a function that defines, at every point of the domain, a target size for the elements at that point. The present ways of defining such a mesh size field in Gmsh are:

1. mesh sizes prescribed at model vertices and interpolated linearly on model edges;

```
class GEdge : public GEntity{
  // bi-directional data structure
  modelVertex *v1, *v2;
  std::list<GFace*> faces;
public:
  // pure virtual functions that have to be overloaded
  // for every solid modeler
  virtual std::pair<double> parRange() = 0;
  virtual Point3 value(double t) = 0;
  virtual Vector3 tangent(double t) = 0;
  virtual Point2 reparam(GFace *mf, double t, int dir) = 0;
  virtual bool isSeam(GFace *mf) = 0;
  // other functions of the class are non pure virtual
  // ...
};
```

Figure B.2: A part of the model edge class description. `GEdge::parRange` returns the range for the parameter in the curve. `GEdge::value` returns the 3-D point $\vec{p}(t)$ that is located on the curve $\mathcal{C}$ for a given parameter t. `GEdge::tangent` evaluates the tangent vector $\partial_t\vec{p}(t)$ for a given parameter t. `GEdge::reparam` computes the local parameters of the point $\vec{p}(t)$ on a model face `mf` that has $\mathcal{C}$ in its closure, `GEdge::isSeam` tells if the curve is or is not a seam of the face `mf`. Generally, seam edges are used to maintain consistency of data structure for periodic surfaces.

```
class GFace : public GEntity{
  // bi-directional data structure
  GRegion *r1, *r2;
  std::list<GEdge*> edges;
public:
  // pure virtual functions that have to be overloaded
  // for every solid modeler
  virtual std::pair<double> parRange(int dir) const = 0;
  virtual Point3 value(double u, double v) const = 0;
  virtual std::pair<Vector3> tangent( double u, double v) const = 0;
  virtual double curvature(double u, double v) const;
  // other functions of the class are non pure virtual
  // ...
};
```

Figure B.3: A part of thet model face class description. `GFace::parRange` returns the range for the parameter in the surface in direction `dir`. `GFace::value` returns the 3-D point $\vec{p}(u,v)$ that is located on the surface $\mathcal{S}$ for the given parameter couple (u,v). `GFace::tangent` the evaluates two tangent vectors $\partial_u\vec{p}(u,v)$ and $\partial_v\vec{p}(u,v)$. The `GFace::curvature` function computes the divergence of the unit normal vector at (u,v). This last function is used for the definition of mesh size fields. It is not a pure virtual function: a default implementation is available using finite differences.

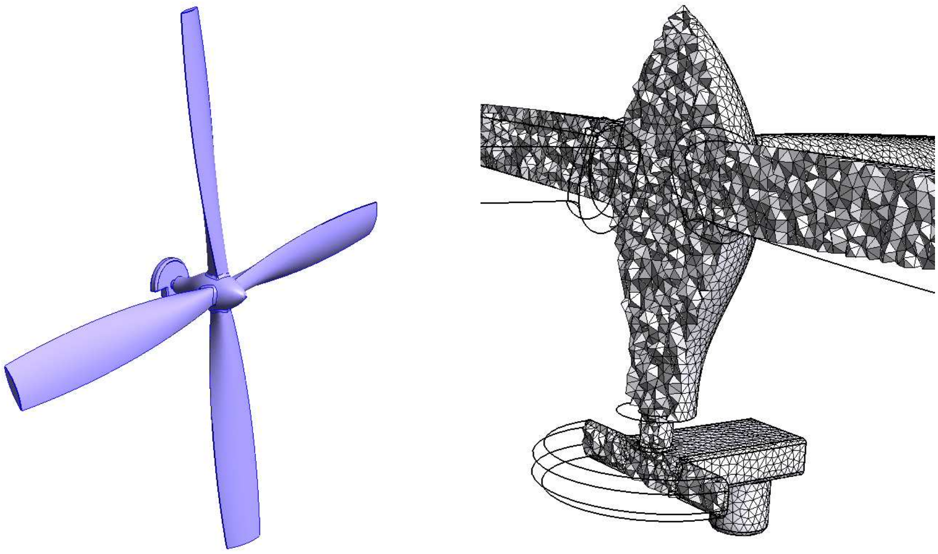


Figure B.4: CAD model of a propeller (left) and its volume mesh (right)

2. prescribed mesh gradings on model edges (geometrical progressions, ...);

3. mesh sizes defined on another mesh (a background mesh) of the domain;

4. mesh sizes that adapt to the principal curvature of model entities.

These size fields can then be acted on by functionals that may depend, for example, on the distance to model entities or on user-prescribed analytical functions; and when several size fields are provided, Gmsh uses the minimum of all fields. Thanks to that mechanism, Gmsh allows for a mesh size field defined on a given model entity to extend in higher dimensional entities. For example, using a distance function, a refinement based on the curvature of a model edge can extend on any surface adjacent to it.

Let us now consider an edge e of the mesh. We define the adimensional length of the edge with respect to the size field δ as

$$l_e = \int_e \frac{1}{\delta(x,y,z)} dl. \tag{B.1}$$

The aim of the mesh generation process is twofold:

1. Generate a mesh for which each mesh edge e is of size close to $l_e = 1$,

2. Generate a mesh for which each element K is *well shaped*.

In other words, the aim of the mesh generation procedure is to be able to build a good quality mesh that complies with the mesh size field.

To quickly evaluate the adequation between the mesh and the prescribed mesh size field, we defined an efficiency index τ [8] as

$$\tau = \exp\left(\frac{1}{ne}\sum_{e=1}^{ne} \tau_e\right) \tag{B.2}$$

with $\tau_e = l_e - 1$ if $l_e < 1$ and $\tau_e = \dfrac{1}{l_e} - 1$ if $l_e \geq 1$. The efficiency index ranges in $\tau \in [0, 1]$ and should be as close as possible to $\tau = 1$.

For measuring the quality of elements, various element shape measures are available in the literature [20, 17]. Here, we choose a measure based on the element radii ratio, i.e. the ratio between the inscribed and the circumcircles.

If K is a triangle, we have the following formula

$$\gamma_K = 4\frac{\sin\hat{a}\sin\hat{b}\sin\hat{c}}{\sin\hat{a} + \sin\hat{b} + \sin\hat{c}},$$

$\hat{a}$, $\hat{b}$ and $\hat{c}$ being the three inner angles of the triangle. With this definition, the equilateral triangle has a $\gamma_K = 1$ and degenerated (zero surface) triangles have a $\gamma_K = 0$.

For a tetrahedron, we have the following formula:

$$\gamma_K = \frac{6\sqrt{6}\,V_k}{\left(\sum_{i=1}^{4} a(f_i)\right) \max_{i=1,\ldots,6} l(e_i)},$$

with V_K the volume of K, $a(f_i)$ the area of the i^{th} face of K and $l(e_i)$ the dimensional length of the i^{th} edge of K. This quality measurement lies in the interval $[0, 1]$, an element with $\gamma_K = 0$ being a sliver (zero volume).

B.6 1-D Mesh Generation

Let us consider a point $\vec{p}(t)$ on a curve $\mathcal{C}$, $t \in [t_1, t_2]$. The number of subdivisions N of the curve is its adimensional length:

$$\int_{t_1}^{t_2} \frac{1}{\delta(x, y, z)} \|\partial_t \vec{p}(t)\| dt = N. \tag{B.3}$$

The $N + 1$ mesh points on the curve are located at coordinates $\{T_0, \ldots, T_N\}$, where T_i is computed with the following rule:

$$\int_{t_1}^{T_i} \frac{1}{\delta(x, y, z)} \|\partial_t \vec{p}(t)\| dt = i. \tag{B.4}$$

With this choice, each subdivision of the curve is exactly of adimensional size 1, and the 1-D mesh exactly satisfies the size field δ. In Gmsh, (B.4) is evaluated with a recursive numerical integration rule.

B.7 2-D Mesh Generation

Curved surface shapes designed by CAD systems are usually defined by parametric surfaces, for example, NURBS [21]. Let us consider a model face G_i^2 with its underlying geometry, in this case a surface $\mathcal{S} \in R^3$ with its parametrization $\vec{p}(u, v) \in \mathcal{S}$, where the domain of definition of the parameters (u, v) is defined by a series of boundary curves. An example of such a surface is given in Figure B.5, which shows one of the 76 model faces of the propeller in the parametric space (left) and in real space (right). Three features of surface $\mathcal{S}$, common in CAD descriptions, make its meshing non-trivial:

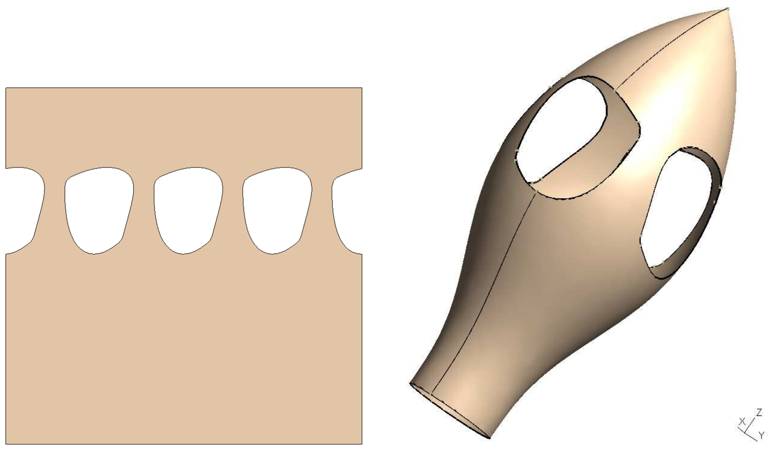


Figure B.5: Geometry of a model face in parametric space (left) and in real space (right). Two seam edges are present in the face. The top model edge is degenerated in one point.

1. $\mathcal{S}$ is periodic. The topology of the model face is modified in order to define its closure properly. A seam is present two times in the closure of the model face. These two occurrences are separated by one period in the parametric space.

2. $\mathcal{S}$ is trimmed: it contains four holes and one of them is crossed by the seam.

3. One of the model edges of $\mathcal{S}$ is degenerated. This is done for accounting of a singular point in the parametrization of the surface. This kind of degeneracy is present in many shapes: spheres, cones and other surfaces of revolution.

Techniques for generating finite element meshes on curved surfaces are of two kind:

1. techniques for which the surface mesh is generated directly in the real 3-D space;

2. techniques for which the surface mesh is generated in the parametric space.

The principal advantage of doing the surface mesh in the 3-D space directly is that no interface to the solid modeler is required. Such algorithms have been used for building meshes from STL (stereolithography) data files [1], from medical imaging [32] or to adapt/modify existing meshes [6]. The main drawback of such algorithms is their relative lack of robustness.

The second alternative can be applied only if a parametrization of the surfaces is available. If it is the case, doing the mesh in the parametric space is usually advantageous because all the meshing procedures can be applied in the parametric plane. This allows mesh operators to be highly robust—we will detail that argument later.

Yet, all surfaces do not always have a parametrization that conserves angles and lengths. Consequently, only 2-D algorithms that allow to build anisotropic meshes in the plane can be considered as good candidates for doing surface meshing. Figure B.6 presents the surface mesh of the model face of Figure B.5, both in the parametric space and in the real space. The mesh in the real space is isotropic and uniform while the one in the parametric space is highly anisotropic and non uniform. To solve this problem, George and Borouchaki [7] have proposed the use of a metric derived from the first fundamental form of the surface. The metric field

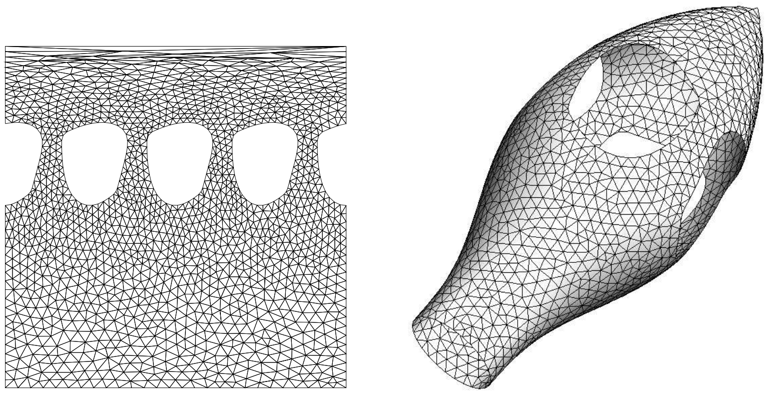


Figure B.6: Mesh of a model face drawn in the parametric space (left) and in the real space (right).

is a second order tensor field that has the form, at any point of the parametric space, of a 2×2 matrix. The metric is used to define angles and distances in parametric space. With their Delaunay approach, the "empty circle" property, effectively becomes an "empty ellipse" property. An equivalent "metric-based" advancing front surface mesh generation algorithms is presented by Cuilliere in [3]. A more exotic metric-based approach based on packing ellipses has been devised by Yamada et al. [28] and has been used more recently by Lo and Wang in [18].

In addition to a Delaunay implementation similar to [7] and a frontal-Delaunay meshing technique inspired by [22], Gmsh provides an original surface meshing strategy based on the concept of local mesh modifications [15, 16, 23]. The main advantage of the new approach, compared to the other ones based on the Delaunay criterion, is that it does not require the computation of derivatives of the parametrization. For that reason, the new approach remains robust even when the parametrization is singular.

The algorithm works as follows. First, an initial mesh containing all the mesh points of the curves bounding the face is built in the parametric space using a divide and conquer strategy [4]. Then, all the edges of the 1-D discretization are recovered using swaps [31]. Finally, local mesh modifications are applied:

1. Each edge that is too long is split;

2. Each edge that is too short is removed using an edge collapse operator;

3. Edges for which a better configuration is obtained by swapping are swapped;

4. Vertices are re-located optimally.

More precisely, here is how these four local mesh modifications procedures are applied in Gmsh:

Edge Splitting: An edge is considered too long when its adimensional length is greater than $l_e > 1.4$. When split, the two new edges will have a minimal size of 0.7. In order to converge to a stable configuration, an edge of size $l_e = 0.7$ should not be considered as a short edge.

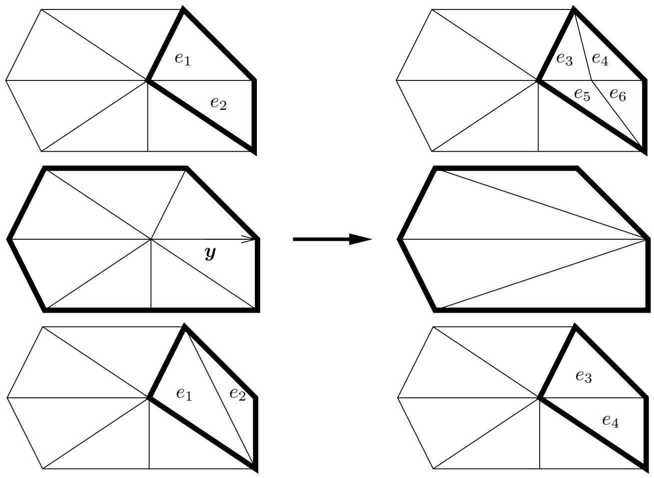


Figure B.7: Illustration of local mesh modifications.

Edge Collapsing: An edge is considered to be short when its adimensional length is smaller than $l_e < 0.7$. An edge cannot be collapsed if one of the remaining triangles after the collapse is inverted in the parametric space.

Edge Swapping: An edge is swapped if $\min(\gamma_{e_1}, \gamma_{e_2}) < \min(\gamma_{e_3}, \gamma_{e_4})$ (see Figure B.7), unless

1. it is classified on a model edge;
2. the two adjacent triangles e_1 and e_2 form a concave quadrilateral in the parametric space;
3. the angle between the triangles normals is greater than a threshold, typically 30 degrees.

Vertex Re-positioning: Each vertex is moved optimally inside the cavity made of all its surrounding triangles. The optimal position is chosen in order to maximize the worst element quality [5].

For each of these local mesh modification procedures, the *opportunity* of doing a mesh modification is evaluated in the real space, i.e., in (x, y, z), while the *validity* of a mesh modification is evaluated in the parametric space (u, v). Therefore, Gmsh mesh generators always retain both real and parametric coordinates of any mesh vertex. To ensure robustness, all the elementary geometrical predicates make use of robust algorithmics [27].

In practice, this algorithm converges in about 6-8 iterations and produces anisotropic meshes in the parametric space without computing derivatives of the mapping.

Let us illustrate the algorithm on an example. We have meshed the model face of Figure B.6 using an analytical size field

$$\delta(x, y, z) = \delta_0[1 + \cos(\pi(x + y - z)/L)] + \epsilon$$

where L is a characteristic size of the domain and $\epsilon \ll \delta_0 < L$. Figure B.8 shows the mesh in the parametric space at different stages of the algorithm. Note that the derivatives of the

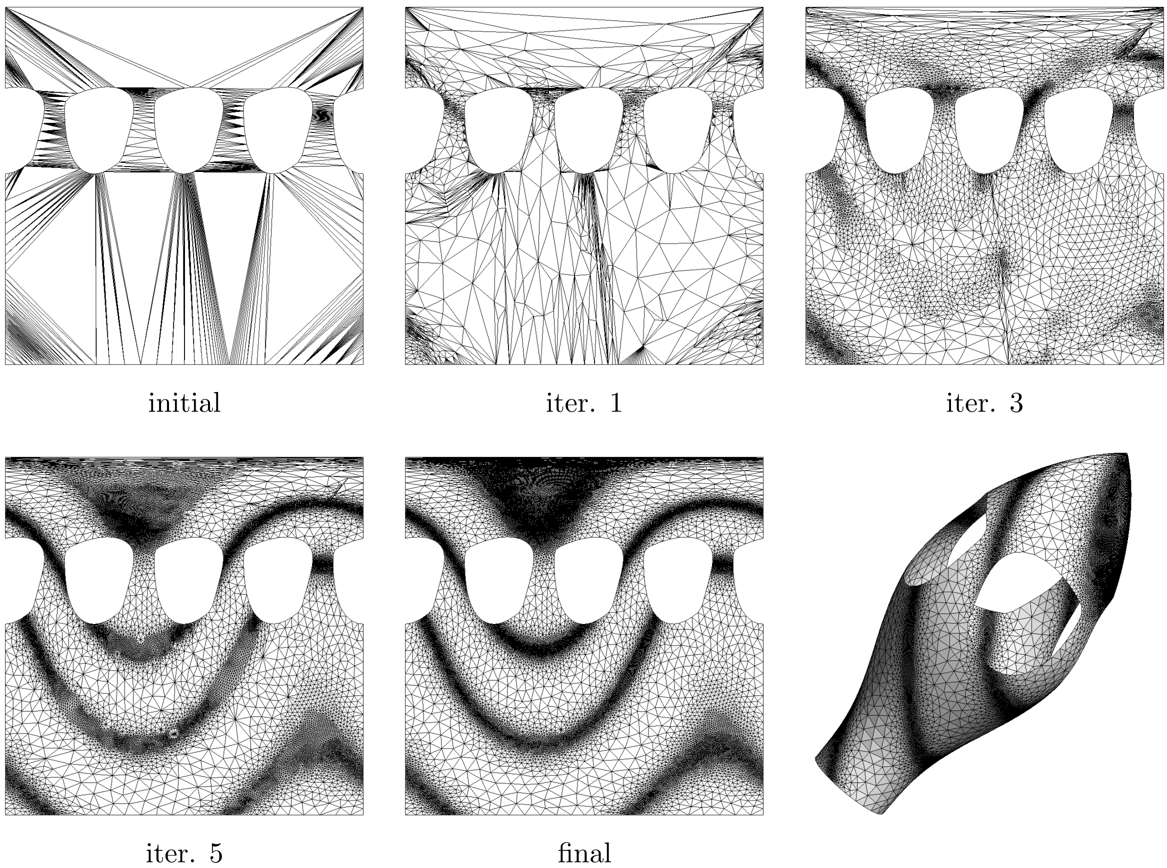


Figure B.8: Illustration of the surface meshing algorithm based on local mesh modifications. The images correspond to the initial mesh containing boundary vertices, the mesh after 1, 3, 5 and 8 iterations. At iteration 8, the algorithm has converged. The size field efficiency $\tau = 0.89$ can be considered as excellent: 90 % of the elements have a radii ratio γ_K greater that 0.9.

parametrization of the underlying surface are not defined at the singular point so that any algorithm that requires to compute such derivatives would be in trouble in this case.

B.8 3-D Mesh Generation

Once a surface triangulation is available, an automatic mesh generation procedure does not usually require an interface to a CAD system. Indeed, Gmsh interfaces several open source 3-D tetrahedral mesh generation kernels [25, 29] in addition to its own Delaunay refinement algorithm. These algorithms are standard [9, 25] and will not be explained here. We focus on two other issues instead:

1. the way Gmsh interfaces multiple mesh generation algorithms;
2. the way Gmsh optimizes the quality of 3-D meshes.

(A third issue concerns the way Gmsh handles mixed structured/unstructured grids—this is addressed in Section B.10.)

B.8.1 Mesh Algorithm Interfaces

Gmsh is able to deal with most of the standard finite element shapes: lines, triangles, quadrangles, tetrahedra, hexahedra, prisms and pyramids. The internal mesh data structures are designed to minimize the memory footprint without compromising flexibility: in addition to a integer tag and a partition/visualisation index, any element only holds its ordered list of vertices. With that simple design, Gmsh can load about 12 million tetrahedra per Gigabyte of memory (28 bytes per tetrahedron, 44 bytes per mesh vertex), including graphics representation, i.e., OpenGL vertex arrays [26].

When "in house" meshing routines are used, Gmsh derives (in the object oriented sense) enriched data structures specifically tailored for each meshing algorithm. Those derived structures contain just the right extra information necessary: the parametric coordinates of a vertex for parametric 2-D meshing algorithms, the neighbours of a tetrahedron for the 3-D Delaunay algorithm, etc. With this approach the footprint of a tetrahedron is for example extended to 84 bytes, and the 3-D Delaunay algorithm implemented in Gmsh, using a classical Bowyer-Watson algorithm [30], is able to build about 7 million tetrahedron per Gigabyte of memory (including overhead like the data structures of the CAD engine).

When a third party mesh generator is invoked, Gmsh needs of course to allocate the appropriate structures required by that specific software. But thankfully, while transferring the data from the third party algorithm into Gmsh, only the minimal internal data structures need to be allocated (i.e., 28 byte per tetrahedron in the 3-D case mentioned above). This greatly reduces the overhead incurred by interfacing external algorithms.

B.8.2 Tetrahedral Mesh Improvement

Tetrahedral mesh improvement is usually required to produce 3-D meshes suitable for grid-based numerical methods [5]. Unfortunately, mesh optimization procedures have a lot to do with "black magic": even if the ingredients required to construct a mesh optimization procedure are well known (essentially swapping and smoothing), there is no known "best recipe", i.e., no known optimal way of combining those smoothing and swapping operators.

Gmsh implements its own mesh optimization procedure to enhance tetrahedral mesh quality by means of edge- and face-swappings and vertex relocations, and also interfaces third party mesh optimizers—in particular the open-source optimizer from Netgen [25]. Interestingly, applying optimization routines one after the other enables to produce better meshes than applying mesh optimizers separately. Figure B.9 shows the distribution of elemental qualities on the mesh of a toroidal domain containing about 600,000 tetrahedra (the mesh was generated in about 30 seconds with the "in house" 3-D Delaunay algorithm). The unoptimized mesh contains quite a few ill shaped elements: more than 5000 elements have an aspect ratio γ_K below 0.2 and the worst shaped element has an aspect ratio of 10^{-3}. After one pass of the Gmsh mesh optimizer, which takes about 12 seconds, all slivers have disappeared and the worst element has an aspect ratio of 0.32. The distribution of element quality is enhanced, with a clear right shift of the distribution. Applying the Netgen optimizer after the Gmsh optimizer, additional improvement can be observed: the worst elemental quality is now 0.41 and another shift to the right has occurred. However, the application of the

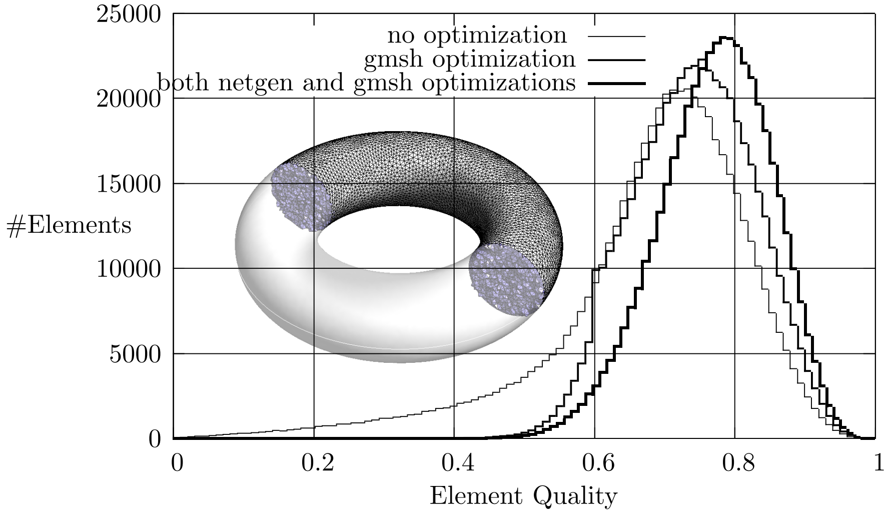


Figure B.9: Distribution of γ_K in a mesh of about $600,000$ tetrahedra.

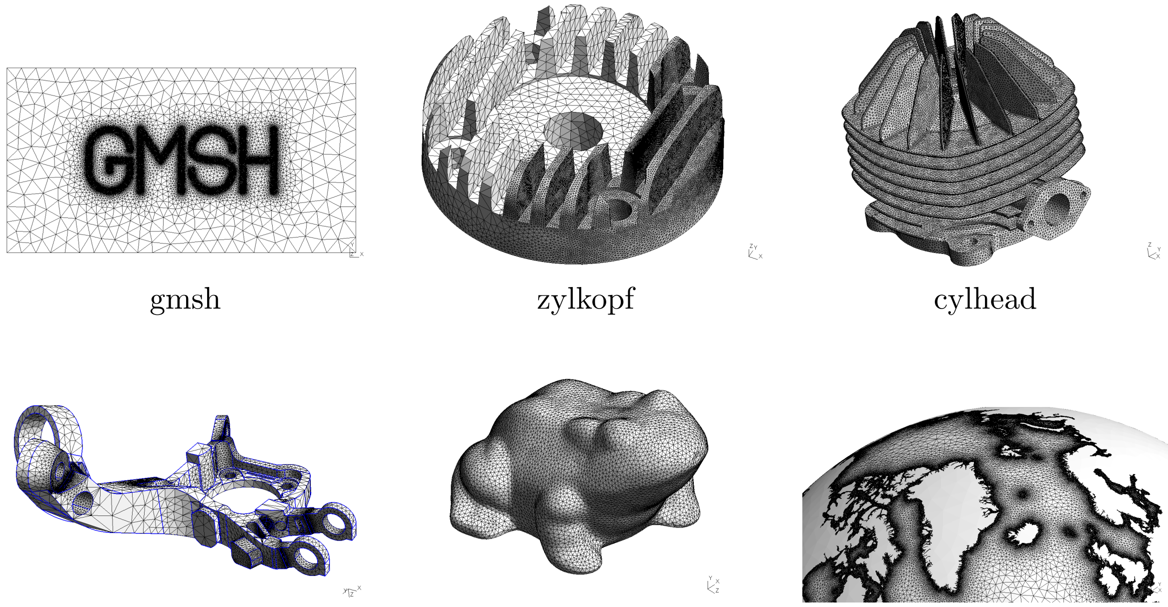
Netgen optimizer also dramatically reduced the number of elements in the mesh, and this second optimization pass took more than 200 seconds—about 15 times more than for the Gmsh optimizer. Transferring the mesh in Netgen format also doubled the memory usage.

B.9 Examples

One of the objectives of this section is to demonstrate that Gmsh is able build meshes that can be used by the finite element community. The various examples shown below can all be downloaded from the Gmsh web site. They come from different sources: native Gmsh CAD models, CAD models found on the web, or CAD models that were proposed by industrial and academic partners. The formats considered are IGES, STEP, BREP and Gmsh. Various size fields have been used, uniform or not. Both 2-D and 3-D statistics are provided.

Table B.1 presents details for some of the models (see Figure B.10) used in the Gmsh 2-D test suite. Mesh size field are defined in a variety of ways: analytic, uniform, size fields driven by distance functions (attractors), boundary layers, size fields related to the curvature of surfaces, or size fields interpolated using sizes that are defined on model vertices. Table B.2 gives statistics for the 2-D meshes generated with the surface meshing algorithm presented in Section B.7. In the case of planar surfaces and uniform meshes this algorithm is about three times slower than the 2-D anisotropic Delaunay mesh generator implemented in Gmsh. However, when multiple size fields are involved and/or when the surfaces are very complex, this new approach becomes competitive in terms of CPU time—and is much more robust than the anisotropic Delaunay mesher. With the caveat that the performance and robustness of mesh generation algorithms are highly dependent on their implementation, we believe this shows evidence that the new algorithm proposed in Section B.7 is a viable alternative to classical Frontal or Delaunay approaches.

Table B.3 presents some statistics for 3-D meshes. The first example can serve as reference: it is a unit cube that is meshed uniformely with about one million tetrahedra. Some of the examples have complex mesh size fields (linkrods or frogadapt). One has small features in



gmsh zylkopf cylhead

fuse frog world ocean

Figure B.10: Some images of surface meshes.

	type	n_R	n_F	n_E	n_V	δ
cube	GMSH	1	6	12	8	uniform
gmsh	GMSH	0	1	35	23	attractor
frog	IGES	1	475	950	477	uniform
frogadapt	IGES	1	475	950	477	analytic
linkrods	STEP	1	37	108	74	analytic
zylkopf	STEP	1	137	404	270	at vertices
cylhead	BREP	1	1054	2485	1445	uniform
fuse	STEP	1	249	723	476	curvature
block	STEP	1	533	1586	1048	uniform
senzor	GMSH	8	90	200	146	at vertices
world ocean	GMSH	0	1	4245	145291	boundary layer
media	GMSH	1274	8398	5779	3894	uniform

Table B.1: Statistics on the models that are considered: n_R,n_F,n_E and n_V are respectively the number of model regions, of model faces, of model edges and of model vertices in the model. δ is the size field.

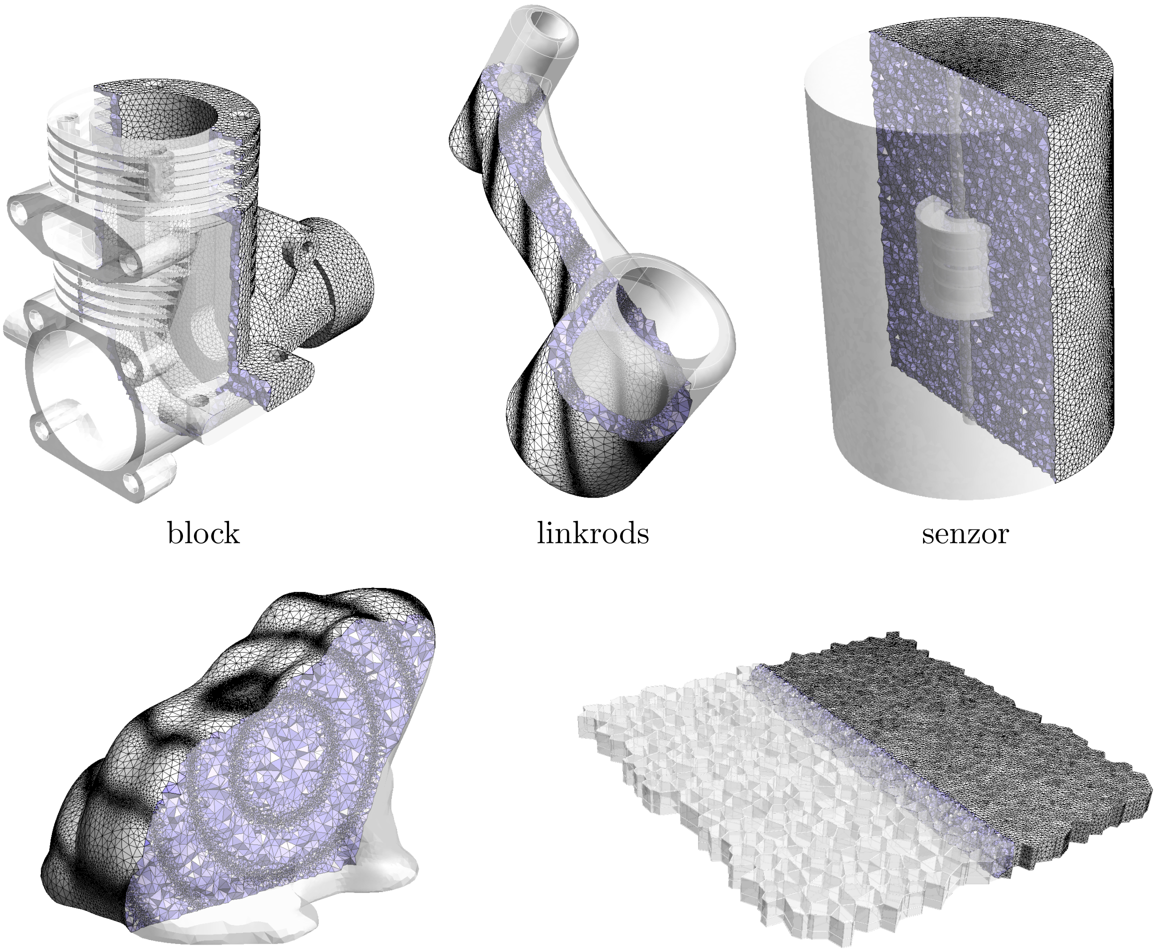


Figure B.11: Some images of volume meshes.

	np	ne	$\gamma_K > 0.9$	$\min_K \gamma_K$	$\text{avg}_K \gamma_K$	$l_{\sqrt{2}}$	τ	CPU
gmsh	28041	55922	83.5%	0.287	0.946	99.0%	0.891	10 s
linkrods	55959	119922	84.2%	0.385	0.946	98.9%	0.893	61 s
zylkopf	32806	65668	86.0%	0.105	0.947	98.5%	0.860	8 s
cylhead	84014	188150	77.5%	0.050	0.915	95.5%	0.892	45 s
fuse	23485	47038	76.0%	0.010	0.919	97.3%	0.886	11 s
block	19694	55530	76.0%	0.021	0.923	96.8%	0.895	20 s
senzor	19876	40002	84.6%	0.546	0.947	98.4%	0.896	11 s
ocean	1152011	2255212	89.0%	0.211	0.950	99.1%	0.901	729 s

Table B.2: Surface mesh generation statistics. Here, np are ne are the number of points and triangles in the surface mesh, $\gamma_K > 0.9$ states for the percentage of triangles that have a quality measure γ_K greater that 0.9, $\min_K \gamma_K$ is the worst element quality in the surface mesh and $\text{avg}_K \gamma_K$ is the average elemental quality. The quantity $l_{\sqrt{2}}$ states for the percentage of edges that have an adimensional length in the range $1/\sqrt{2} < l_e < \sqrt{2}$. The factor τ is the efficiency index defined in Equation (B.2). The last column gives the CPU time (in seconds) for performing the surface mesh generation.

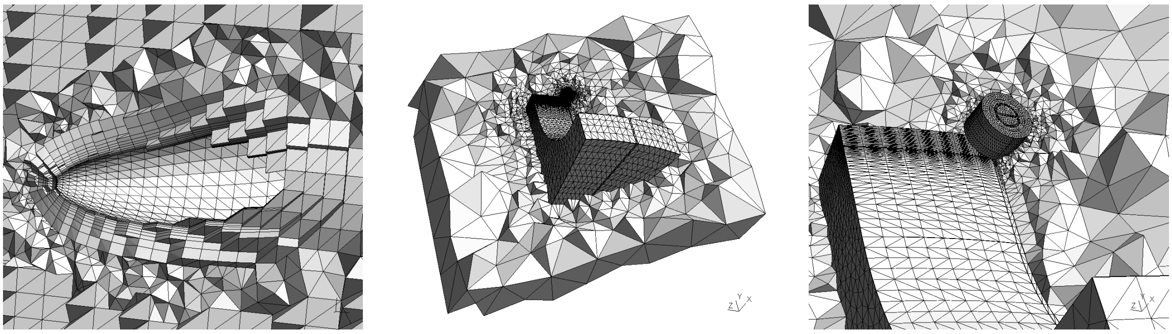
	np	ne	$\min_K \gamma_K$	$\text{avg}_K \gamma_K$	CPU (mesh)	CPU (opti)
cube	195,671	1,098,530	0.235	0.717	49 sec.	36 s
linkrods	341,297	1,836,634	0.347	0.756	111 s	117 s
block	48,897	221,090	0.012	0.660	12 s	14 s
senzor	143,799	805,392	0.222	0.765	35 s	27 s
frogadapt	403,947	2,381,969	0.172	0.691	116 s	180 s
media	164,517	890,756	0.071	0.696	55 s	31 s

Table B.3: Volume mesh generation statistics. Here, np are ne are the number of points and tetrahedron in the volume mesh, $\min_K \gamma_K$ is the worst element quality in the volume mesh and $\text{avg}_K \gamma_K$ is the average elemental quality. The last two columns give mesh generation and mesh optimization timings.

the geometry (block). Some have multiple volumes (media or senzor). Complex mesh size fields such as the ones of linkrods or frogadapt make the mesh generation process slower of about 20%. This overhead is essentially due to the evaluation of the mesh size field. Mesh with strong size variations or with small geometric features require more optimization. The performance figures mentioned in Section B.8.1 hold even for models with a large number of model regions: the model called “media”, created in the native Gmsh CAD format, involves over 1000 model regions and was meshed using the 3-D Delaunay algorithm in less than one minute. All timings were measured on a standard MacBook Pro with a CPU clocked at 2.0 GHz.

B.10 Mixed Meshes

In addition to unstructured meshes, Gmsh enables the generation of simple *structured* meshes in 1-D, 2-D and 3-D, and allows to couple these with unstructured meshes. Structured technologies include transfinite and elliptic meshes as well as a variety of sweeping techniques.



boundary layer sweep sweep (detail)

(R. V. Sabariego, Univ. of Liège)

Figure B.12: Some images of hybrid volume meshes.

For example, to build a boundary layer mesh from a set of source surfaces (see Figure B.12), Gmsh

1. creates the topology of the boundary layer by sweeping zero-height volumes from all the source surfaces. (In addition to the boundary layer volumes, this creates a set of new boundary layer points, curves and surfaces. These new points, curves and surfaces will only acquire a concrete representation during the meshing process.)

2. meshes the source surfaces and computes (unique) normals at the mesh vertices;

3. sweeps the boundary layer points along the normals, and remeshes all the non-boundary-layer curves connected to these points, and then meshes the boundary layer curves by sweeping along the normals;

4. meshes the non-source surfaces, then the boundary layer surfaces (again by sweeping along the normals), and finally the volumes.

Once all the structured parts are meshed, the remaining parts are meshed using the unstructured algorithms, resulting in conforming mixed meshes. In order for the final mesh to be conforming, the structured algorithm splits hexahedra, prisms or pyramids into simplices when needed.

Appendix C

Reparametrization and CAD cleaning

This appendix details the remeshing capabilities of Gmsh, currently under development. Part of the work detailed below was supported by the Walloon Region under WIST2 grant EF-CONIVO.

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2010; **83**:403–425
Published online 18 February 2010 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme.2824

High-quality surface remeshing using harmonic maps

J.-F. Remacle[1,*,†], C. Geuzaine[2], G. Compère[1] and E. Marchandise[1]

[1] *Institute of Mechanics, Materials and Civil Engineering (iMMC), Université catholique de Louvain, Place du Levant 1, 1348 Louvain-la-Neuve, Belgium*
[2] *Department of Electrical Engineering and Computer Science, Université de Liège, Montefiore Institute B28, Grande Traverse 10, 4000 Liège, Belgium*

SUMMARY

In this paper, we present an efficient and robust technique for surface remeshing based on harmonic maps. We show how to ensure a one-to-one mapping for the discrete harmonic map and introduce a cubic representation of the geometry based on curved PN triangles. Topological and geometrical limitations of harmonic maps are also put to the fore and discussed. We show that, with the proposed approach, we are able to recover high quality meshes from both low input STL triangulations and complex surfaces defined by many CAD patches. The overall procedure is implemented in the open-source mesh generator Gmsh.

Received 11 August 2009; Revised 6 November 2009; Accepted 13 November 2009

KEY WORDS: surface remeshing; surface parametrization; STL file format; surface mapping; harmonic map; surface smoothing

1. INTRODUCTION

Creating high-quality meshes is an essential feature for obtaining accurate and efficient numerical solutions of partial differential equations as it impacts both the accuracy and the efficiency of the numerical method using those meshes [1, 2].

In many cases, surfaces do not have a standard CAD representation and are only known by triangulations such as stereolithography (STL) triangulations. These kinds of surfaces are commonplace in many areas of science and engineering, e.g. in the form of 3D scanned images, terrain data, or medical data obtained from imaging techniques through a segmentation procedure. Such triangulations are often oversampled and/or of poor quality (with triangles exhibiting very

*Correspondence to: J.-F. Remacle, Institute of Mechanics, Materials and Civil Engineering (iMMC), Université catholique de Louvain, Place du Levant 1, 1348 Louvain-la-Neuve, Belgium.
†E-mail: jean-francois.remacle@uclouvain.be

Contract/grant sponsor: Fonds National de la Recherche Scientifique, rue d'Egmont 5, 1000 Bruxelles, Belgium

small aspect ratios), which makes them unsuited for direct use by numerical methods like finite elements, finite volumes, or boundary elements. This is also problematic for the volume mesh since the surface mesh serves as input for the volume meshing algorithms. Improving the mesh quality can then be performed using a remeshing procedure.

In the case of manufactured objects, the surfaces are often designed using a CAD system and described through a constructive solid geometry procedure. Non Uniform Rational B-Splines (NURBS) are commonly used for describing the shape of surfaces. NURBS surfaces are usually nice and smooth so that it is possible to produce high-quality surface meshes using NURBS as input. However, most surface mesh algorithms mesh model faces individually, which means that points are generated on the bounding edges and that these points will be part of the surface mesh. If thin CAD patches exist in the model they will result in the creation of small distorted triangles with very small angles [3, 4]—even if the bounding edges of these thin patches have no physical significance. As in the case of a poor quality STL triangulation, a remeshing procedure is also then desirable.

There are mainly two approaches for surface remeshing: mesh adaptation strategies [5–7] and parametrization techniques [8–13]. Mesh adaptation strategies use local mesh modifications in order both to improve the quality of the input surface mesh and to adapt the mesh to a given mesh size criterion. In parametrization techniques, the input mesh serves as a support for building a continuous parametrization of the surface. (In the case of CAD geometries, the initial mesh can be created using any off-the-shelf surface mesher for meshing the individual patches.) Surface parametrization techniques originate mainly from the computer graphics community: they have been used extensively for applying textures onto surfaces [14, 15] and have become a very useful and efficient tool for many mesh processing applications [16–20]. In the context of remeshing procedures, the initial surface is parametrized onto a surface in $\mathscr{R}^2$, the surface is meshed using any standard 2D mesh generation procedure and the new triangulation is then mapped back to the original surface [3, 21].

This paper proposes a quality remeshing strategy based on *harmonic maps* for the surface parametrization (see [20] for a survey of alternative parametrization techniques). Harmonic maps exhibit several useful properties: (i) they are easy to compute and can be approximated using linear systems, (ii) they are independent of the initial triangulation, (iii) they are indefinitely differentiable on a surface, and (iv) they are one-to-one for convex mapped regions [20, 22]. Harmonic maps do not preserve angles such as the conformal maps usually used for texture mapping [15, 18] and by some authors for surface remeshing [23]. However, this is not quite an issue in the context of mesh generation. Indeed, we can deal with non-conforming maps as soon as we have access to the metric tensor that allows us to measure both lengths and angles in the parameter plane.

Discrete harmonic maps have first been successfully used for surface remeshing by Eck [21] and Marcum [3]. However, as mentioned by Floater in [24], discrete harmonic maps are in general not guaranteed to be one-to-one. To ensure a one-to-one discrete map, Floater suggested a different edge spring weighting that guarantees an embedding for convex boundaries, also called 'convex combination map'. We show however in this paper that the quality of the metrics of the convex combination map are not sufficient for generating high-quality meshes and suggest to only locally apply a simple geometrical algorithm called 'cavity check'. Another important but rarely discussed issue regarding harmonic maps concerns the geometrical aspect of the surfaces to be parametrized. By presenting the harmonic maps as the solution of Laplace equations, we show why the harmonic mapping fails for surfaces with large aspect ratio. We then suggest some ways to address this issue.

The aim of the paper is twofold: (i) we first present the harmonic mapping in a comprehensive manner such that it becomes accessible to a wider community than the one of computer graphics and (ii) we show that even with the known limitations of harmonic maps, they can be used for efficiently generating high-quality surface meshes. The paper also deals with implementation. We show a simple way to compute and implement efficiently harmonic maps using linear finite elements with appropriate boundary conditions. We show how to guarantee that the discrete harmonic mapping is one-to-one and well-defined for geometries with large aspect ratios. The remeshing procedure is enhanced by using cubic mapping to smooth the initial triangulation. Finally, different results demonstrate that high-quality unstructured meshes can be efficiently and consistently generated for subsequent numerical simulations. We show that the resulting surface meshes that are produced with the new technique have a better quality than standard available remeshing techniques.

All the results presented in the paper were generated using the open-source mesh generator Gmsh [25], where the proposed algorithms can be further studied, tested, and enhanced.

2. PARAMETRIZATION OF DISCRETE SURFACES

Parametrizing a surface $\mathscr{S}$ is defining a map $\mathbf{u}(\mathbf{x})$

$$\mathbf{x} \in \mathscr{S} \subset \mathscr{R}^3 \mapsto \mathbf{u}(\mathbf{x}) \in \mathscr{S}' \subset \mathscr{R}^2 \tag{1}$$

that transforms continuously a 3D surface $\mathscr{S}$ into a surface $\mathscr{S}'$ embedded in $\mathscr{R}^2$ that has a well-known parametrization (see Figure 1). Such a continuous parametrization exists if the two surfaces $\mathscr{S}$ and $\mathscr{S}'$ have the same topology, that is have the same genus $G(\mathscr{S})$ and the same number of boundaries N_B. The genus $G(\mathscr{S})$ of a surface is the number of handles in the surface. For example, a sphere has a genus $G=0$ and $N_B=0$, a disk has $G=0$ but $N_B=1$, and a torus has $G=1$ and $N_B=0$.

In this work, we consider that the only available representation of a surface $\mathscr{S}$ is a conforming triangular mesh $\mathscr{S}_{\mathscr{T}}$ in 3D , i.e. the set of triangles T_j that intersect only at common vertices or

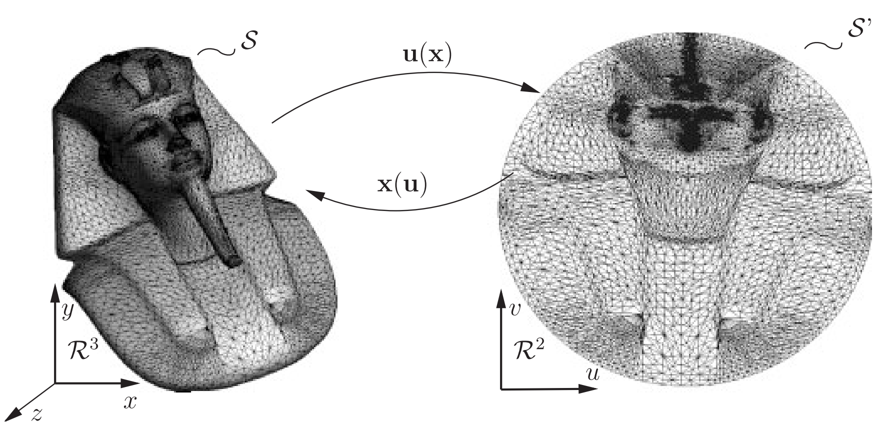


Figure 1. Parametrization $\mathbf{x}(\mathbf{u})$ and inverse parametrization $\mathbf{u}(\mathbf{x})$ of the Tutankhamun mask that associates every point of the surface $\mathscr{S} \subset \mathscr{R}^3$ with a point of the surface $\mathscr{S}' \subset \mathscr{R}^2$.

Int. J. Numer. Meth. Engng 2010; **83**:403–425
DOI: 10.1002/nme

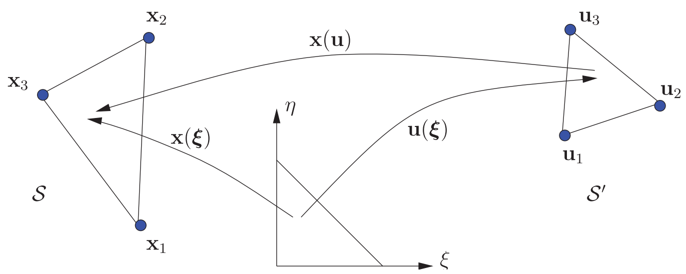


Figure 2. Unit triangle in local coordinates and the maps $\mathbf{x}(\xi), \mathbf{u}(\xi)$, and $\mathbf{x}(\mathbf{u})$.

edges $\mathscr{T} = \{T_1, \ldots, T_N\}$. Let us consider a triangulated surface $\mathscr{S}$ that has N_V vertices, N_E edges, and N_T triangles. The genus $G(\mathscr{S}_{\mathscr{T}})$ is given through the Euler-Poincaré formula:

$$G(\mathscr{S}_{\mathscr{T}}) = \frac{-N_V + N_E - N_T + 2 - N_B}{2}. \tag{2}$$

As an example, the left part of Figure 1 shows a triangulated Tutankhamun mask. This triangulated surface is homeomorphic to the unit disk, i.e. they have both a zero genus and one boundary. It is therefore possible, in principle, to find a smooth transformation that maps $\mathscr{S}$ into $\mathscr{S}'$.

The parametrization we look for is discrete: each vertex V_i, $i = 1, \ldots, N_V$ of the triangulation has two sets of coordinates: the 3D coordinates $\mathbf{x}_i = (x_i, y_i, z_i) \in \mathscr{S}$ and the parametric coordinates $\mathbf{u}_i = (u_i, v_i) \in \mathscr{S}'$. Each triangle also has two representations, one in the 3D space and one in the 2D parametric space. Consider triangle T_j with its three vertices V_1, V_2, and V_3. The parametrization is one-to-one if and only if triangles do not overlap in the parametric space. Note that the notion of triangle overlapping is only well defined in a 2D space.

This triangle can itself be parametrized using for example barycentric coordinates, i.e. standard finite element linear shape functions (see Figure 2):

$$\mathbf{x}(\xi) = (1 - \xi - \eta)\mathbf{x}_1 + \xi \mathbf{x}_2 + \eta \mathbf{x}_3. \tag{3}$$

Similarly, we can also parametrize the triangle in $\mathscr{S}'$:

$$\mathbf{u}(\xi) = (1 - \xi - \eta)\mathbf{u}_1 + \xi \mathbf{u}_2 + \eta \mathbf{u}_3. \tag{4}$$

In order to compute the mapping $\mathbf{x}(\mathbf{u})$, we first invert (4):

$$\mathbf{u} - \mathbf{u}_1 = \underbrace{\begin{bmatrix} u_2 - u_1 & u_3 - u_1 \\ v_2 - v_1 & v_3 - v_1 \end{bmatrix}}_{\mathbf{u}_{,\chi}} \underbrace{\begin{pmatrix} \xi \\ \eta \end{pmatrix}}_{\chi} \tag{5}$$

which gives

$$\xi(\mathbf{u}) = (\mathbf{u}_{,\xi})^{-1}(\mathbf{u} - \mathbf{u}_1) = (\xi_{,\mathbf{u}})(\mathbf{u} - \mathbf{u}_1). \tag{6}$$

The discrete mapping $\mathbf{x}(\mathbf{u})$ can therefore be computed in three steps:

1. Find the unique triangle T_j of the parametric space $\mathscr{S}'$ that contains point $\mathbf{u}$;
2. Compute local coordinates $\xi = (\xi, \eta)$ of point $\mathbf{u}$ inside triangle T_j using Equation (6);
3. Use Equation (3) to compute the mapping $\mathbf{x}(\mathbf{u}) = \mathbf{x}(\xi(\mathbf{u}))$.

Int. J. Numer. Meth. Engng 2010; **83**:403–425
DOI: 10.1002/nme

Mesh generation procedures usually not only require the mapping $\mathbf{x}(\mathbf{u})$ but also its derivatives $\mathbf{x}_{,\mathbf{u}}$. We have

$$\mathbf{x}_{,\mathbf{u}}=\mathbf{x}_{,\xi}\xi_{,\mathbf{u}}=\begin{bmatrix} x_2-x_1 & x_3-x_1 \\ y_2-y_1 & y_3-y_1 \\ z_2-z_1 & z_3-z_1 \end{bmatrix} \frac{\begin{bmatrix} v_3-v_1 & -(u_3-u_1) \\ -(v_2-v_1) & u_2-u_1 \end{bmatrix}}{(u_2-u_1)(v_3-v_1)-(v_2-v_1)(u_3-u_1)}. \tag{7}$$

The metric tensor (or first fundamental form)

$$\mathbf{M}=\mathbf{x}_{,\mathbf{u}}^{\mathrm{T}}\mathbf{x}_{,\mathbf{u}} \tag{8}$$

then allows to compute lengths, angles, and areas. Consider one curve $\mathscr{C}$ drawn on the parametric space $\mathscr{S}'$. Its length is

$$l_{\mathscr{C}}=\int_{\mathscr{C}} \mathrm{d}l=\int_{\mathscr{C}} \sqrt{\mathbf{dx}^2}=\int_{\mathscr{C}} \sqrt{\left(\mathbf{x}_{,\mathbf{u}}\mathbf{du}\right)^2}=\int_{\mathscr{C}} \sqrt{\mathbf{du}^{\mathrm{T}}\mathbf{M}(\mathbf{u})\mathbf{du}}. \tag{9}$$

The practical case for mesh generation is when $\mathscr{C}$ is a mesh edge of the parametric space going from point $\mathbf{u}_1$ to point $\mathbf{u}_2$. Calling $\mathbf{e}=\mathbf{u}_2-\mathbf{u}_1$, its parametrization is

$$\mathscr{C}=\{\mathbf{u}\in\mathscr{S}'|\mathbf{u}=\mathbf{u}_1+t\mathbf{e}, t\in[0,1]\}.$$

In this special case, the length of a straight edge in the parameter space is computed as

$$l_{\mathscr{C}}=\int_0^1 \sqrt{\mathbf{e}^{\mathrm{T}}\mathbf{M}(\mathbf{u}_1+t\ \mathbf{e})\mathbf{e}}\,\mathrm{d}t. \tag{10}$$

3. HARMONIC MAPS WITH APPROPRIATE BOUNDARY CONDITIONS

As illustrated in Figure 1, we have chosen to map our 3D surfaces $\mathscr{S}$ onto a unit disk $\mathscr{S}'$. Therefore, we require genus zero surfaces that have at least one boundary that will be mapped on the unit disk. We compute coordinates u and v separately as solutions of the following two Laplace problems:

$$\begin{aligned} \nabla^2 u &= 0, \quad \nabla^2 v=0 \quad \text{on } \mathscr{S}, \\ u &= \bar{u}(\mathbf{x}), \quad v=\bar{v}(\mathbf{x}) \quad \text{on } \partial\mathscr{S}_1, \\ \partial_n u &= 0, \quad \partial_n v=0 \quad \text{on } \partial\mathscr{S}/\partial\mathscr{S}_1. \end{aligned} \tag{11}$$

Then, we have to supply functions $\bar{u}(\mathbf{x})$ and $\bar{v}(\mathbf{x})$ that map $\partial\mathscr{S}_1$ onto the unit circle. For that, we choose arbitrarily a starting vertex V_s (see Figure 3) and we compute l_i that is the distance from V_S to V_i along $\partial\mathscr{S}_1$. If L is the total length of $\partial\mathscr{S}_1$, the following boundary conditions

$$u(\mathbf{x}_i)=\cos(2\pi l_i/L), \quad v(\mathbf{x}_i)=\sin(2\pi l_i/L) \tag{12}$$

map $\partial\mathscr{S}_1$ onto the unit circle.

Figure 3 shows both an initial triangular mesh of $\mathscr{S}$ and its map onto the unit disk. The surface $\mathscr{S}$ results from the segmentation of an anastomosis site in the lower limbs, more precisely a

DOI: 10.1002/nme

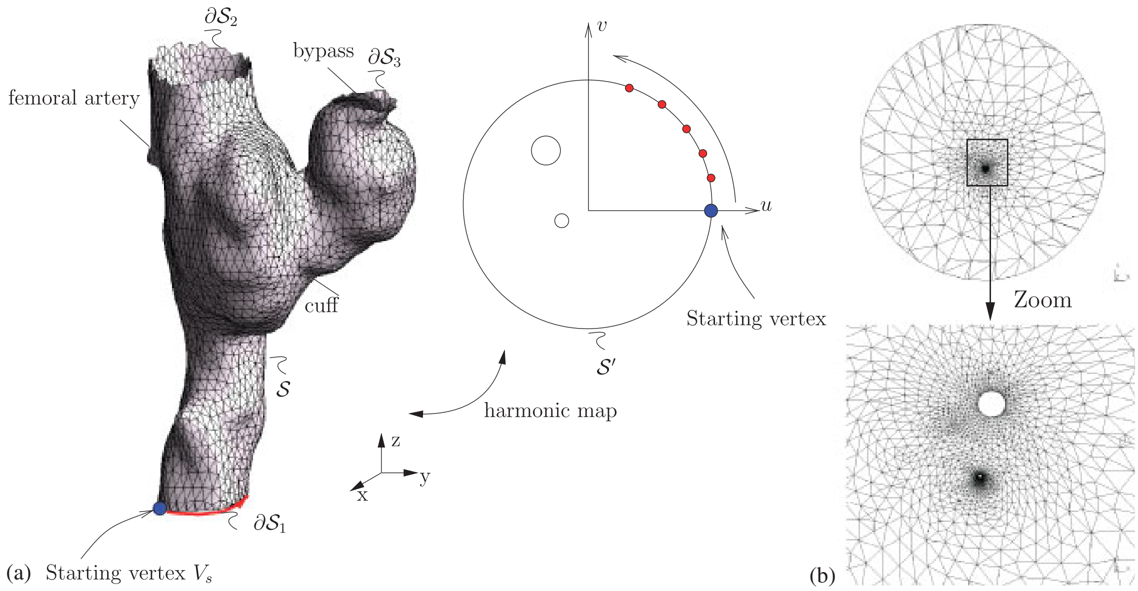


Figure 3. (a) STL triangulation and its map onto the unit disk and (b) the mapped mesh on the unit disk.

bypass of an occluded femoral artery. The unit disk $\mathscr{S}'$ contains two holes that correspond to the boundaries of the femoral artery $\partial\mathscr{S}_2$ and the saphenous vein $\partial\mathscr{S}_3$.

At the continuous level, such a mapping can be proven to be one-to-one, provided that surface $\mathscr{S}'$ is convex. This result is called the Radò-Kneser-Choquet (RKC) theorem [26, 27]. This result strongly depends on the fact that the solution of the Laplace equation obeys a strong maximum principle: $u(\mathbf{x})$ attains its maximum on the boundary $\partial\mathscr{S}$ of the domain. This means that there exists only one single iso-curve $u=u_0$ in $\mathscr{S}$ and that this iso-curve goes continuously from one point of the boundary to another. If another iso-curve $u=u_0$ existed, it should be closed inside $\mathscr{S}$, violating the maximum principle. This is also true for the iso-curve $v=v_0$.

Consider the surface $\mathscr{S}$ of Figure 4(a) with one single boundary ($N_B=1$). Surface $\mathscr{S}'$ is convex if any vertical line $u=u_0$ intersects $\partial\mathscr{S}'$ at most two times. This is also true for any horizontal line $v=v_0$. This means that any coordinate $u=u_0$ ($u_0\in]-1,1[$) appears exactly two times on the boundary $\partial\mathscr{S}$. The two points of $\partial\mathscr{S}$ for which $u=u_0$ are designated as V_A and V_B while the two points for which $v=v_0$ are designated as V_C and V_D. Note that those points appear interleaved while running through $\partial\mathscr{S}$ (V_A appears either after V_D or after V_C but never after V_B). This means that there exists one point in $\mathscr{S}$ for which $u=u_0$ and $v=v_0$.

Now consider the case where $N_B=2$ (Figure 4(b)). Here, zero Neumann boundary conditions are applied to the inner boundary of $\mathscr{S}$. Those are equivalent to the resolution of a Laplace problem on the whole domain while defining a small diffusivity inside the hole (Figure 4(c)). This second problem obeys the same maximum principle as the one with constant diffusivity, which means that the mapping remains one-to-one even when considering holes in the domain.

Of course, any other convex planar surface can serve as $\mathscr{S}'$. In our implementation, we have tried ellipses and rectangles. Yet, no significative difference was observed while changing the definition of the parametric domain.

DOI: 10.1002/nme

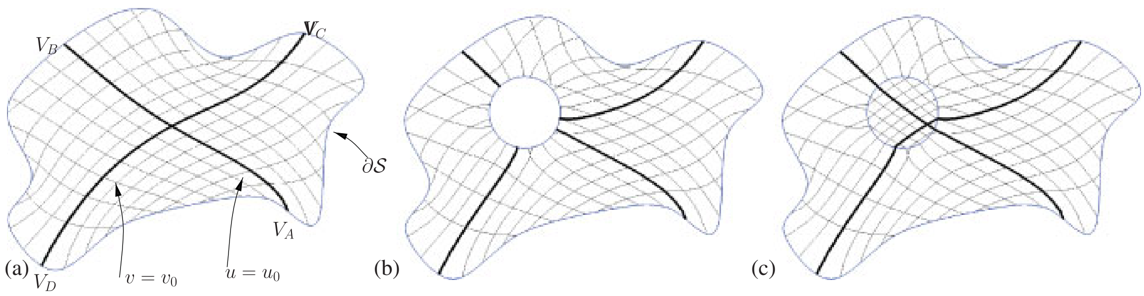


Figure 4. Iso-values of coordinates u and v on a surface $\mathscr{S}$ that are computed as solutions of the Laplace equation on $\mathscr{S}$ with boundary conditions that map $\partial\mathscr{S}$ on the unit circle: (a) Dirichlet boundary conditions are imposed on the outer boundary of $\mathscr{S}$ for two configurations; (b) $\mathscr{S}$ excludes the interior disk and zero Neumann boundary conditions are applied on the inner circular boundary; and (c) $\mathscr{S}$ includes the interior disk, where a small diffusion coefficient is used.

3.1. Discrete harmonic maps with linear finite elements

It is easy to prove that (11) is equivalent to the following quadratic minimization problem:

$$\min_{u\in U(\mathscr{S})} J(u)=\frac{1}{2}\int_{\mathscr{S}} \|\nabla^2 u\|\,\mathrm{d}s \tag{13}$$

with

$$U(\mathscr{S})=\{u\in H^1(\mathscr{S}),\ u=f(\mathbf{x}) \text{ on } \partial\mathscr{S}\}. \tag{14}$$

Assume the following finite expansions for u

$$u_h(\mathbf{x})=\sum_{i\in I} u_i\phi_i(\mathbf{x})+\sum_{i\in J} f(\mathbf{x}_i)\phi_i(\mathbf{x}), \tag{15}$$

where I denotes the set of nodes of $\mathscr{S}_{\mathscr{T}}$ that do not belong to the Dirichlet boundary, J denotes the set of nodes of $\mathscr{S}_{\mathscr{T}}$ that belong to the Dirichlet boundary and where ϕ_i are the nodal shape functions associated to the nodes of the mesh. We assume here that the nodal shape function ϕ_i is equal to 1 on vertex $\mathbf{x}_i$ and 0 on any other vertex: $\phi_i(\mathbf{x}_j)=\delta_{ij}$.

Using the expansion (15), the functional J from (13) can be written as

$$\begin{aligned} J(u_1,\ldots,u_N)=&\frac{1}{2}\sum_{i\in I}\sum_{j\in I} u_i u_j\int_{\mathscr{S}_{\mathscr{T}}}\nabla\phi_i(\mathbf{x})\cdot\nabla\phi_j(\mathbf{x})\,\mathrm{d}s+\sum_{i\in I}\sum_{j\in J} u_i f(\mathbf{x}_j)\int_{\mathscr{S}_{\mathscr{T}}}\nabla\phi_i(\mathbf{x})\cdot\nabla\phi_j(\mathbf{x})\,\mathrm{d}s\\ &+\frac{1}{2}\sum_{i\in J}\sum_{j\in J} f(\mathbf{x}_i)f(\mathbf{x}_j)\int_{\mathscr{S}_{\mathscr{T}}}\nabla\phi_i(\mathbf{x})\cdot\nabla\phi_j(\mathbf{x})\,\mathrm{d}s. \end{aligned} \tag{16}$$

In order to minimize J, we can simply cancel the derivative of J with respect to u_k:

$$\begin{aligned} \frac{\partial J}{\partial u_k}&=\sum_{j\in I} u_j\int_{\mathscr{S}_{\mathscr{T}}}\nabla\phi_j(\mathbf{x})\cdot\nabla\phi_k(\mathbf{x})\,\mathrm{d}s+\sum_{j\in J} f(\mathbf{x}_j)\int_{\mathscr{S}_{\mathscr{T}}}\nabla\phi_k(\mathbf{x})\cdot\nabla\phi_j(\mathbf{x})\,\mathrm{d}s\\ &=0,\quad \forall k\in I. \end{aligned} \tag{17}$$

DOI: 10.1002/nme

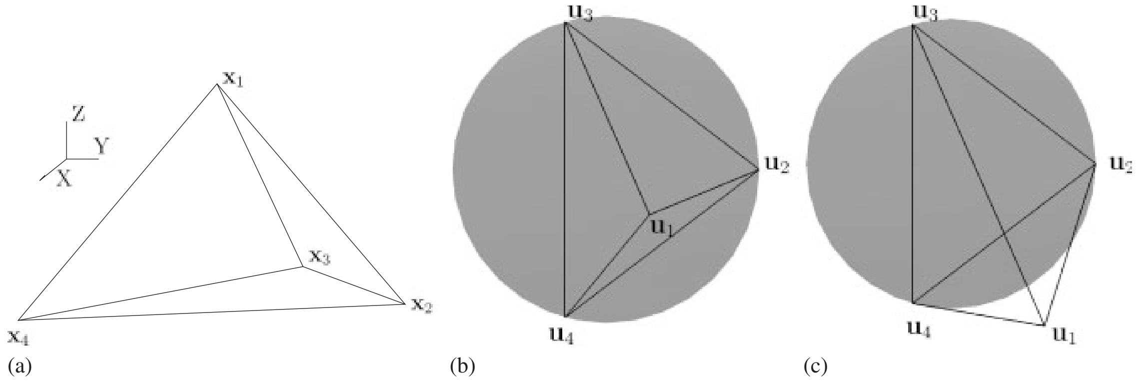


Figure 5. (a) Triangulation for which the discrete harmonic mapping is not guaranteed to be one-to-one: $\mathbf{x}_1=(r,0,1)$ for $r>0$ and $\mathbf{x}_2=(1,1,0)$, $\mathbf{x}_3=(0,0,0)$, $\mathbf{x}_4=(1,-1,0)$; (b) Case $r=1.5$: the mapping is one-to-one; and (c) Case $r=3.5$: the mapping is not one-to-one. The point $\mathbf{u}_1$ does not even lie within the unit disk.

There are as many Equations (17) as there are nodes in I. This system of equations can be proven to be symmetric positive definite so that it can be solved easily, e.g. using preconditioned conjugate gradients. If we want to solve (17) with linear finite elements, we can compute the elementary matrix $A_{ij}^{T_k}$ of triangle T_k as:

$$A_{ij}^{T_k}=\int_{T_k}\nabla\phi_i\cdot\nabla\phi_j\,\mathrm{d}s=\int_0^1\int_0^{1-\xi}\nabla^{\xi,\eta}\phi_i\mathbf{M}_\xi^{-1}\nabla^{\xi,\eta}\phi_j\sqrt{\det\mathbf{M}_\xi}\,\mathrm{d}\xi\,\mathrm{d}\eta, \tag{18}$$

where $\mathbf{M}_\xi$ is the metric tensor of the mapping $\mathbf{x}(\xi)$.

3.2. *One-to-one discrete harmonic map*

In contrast to the continuous harmonic map, it was shown in [24, 28] that the discrete harmonic map first introduced in [21] is not always one-to-one. Indeed, we can see in the next example introduced by Floater [24] that the discrete harmonic map as presented in the previous section is not guaranteed to be one-to-one. Consider a coarse triangulation (Figure 5) made of three triangles $\mathscr{S}_{\mathscr{T}}=\{(1,2,3),(1,3,4),(1,4,2)\}$ and let $\mathbf{x}_1=(r,0,1)$ for some real value $r>0$ and $\mathbf{x}_2=(1,1,0),\mathbf{x}_3=(0,0,0),\mathbf{x}_4=(1,-1,0)$. The three boundary vertices $\mathbf{x}_1,\mathbf{x}_2$, and $\mathbf{x}_3$ are mapped onto the boundary of the unit disk (Figures 5(b) and (c)) and the vertex $\mathbf{x}_1$ should be mapped inside the triangle $(\mathbf{u}_2,\mathbf{u}_3,\mathbf{u}_4)$ to ensure a one-to-one mapping. However, numerical methods for solving Laplace equation may not provide solutions that obey to a discrete maximum principle, especially when meshes are distorted [29], which can lead to discrete harmonic maps that are not one-to-one (Figure 5(c)).

One possibility to ensure a discrete maximum principle consists in placing each point of the parameter plane at the center of gravity of its neighbors. This method was introduced by Floater in [24, 28] and called *convex combination map*. It is implemented simply by choosing

$$A_{ij}^{T_k}=\begin{pmatrix}2&-1&-1\\-1&2&-1\\-1&-1&2\end{pmatrix} \tag{19}$$

DOI: 10.1002/nme

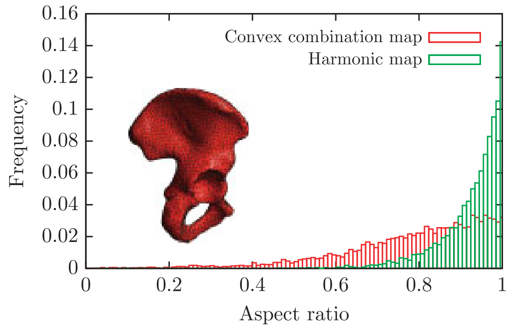


Figure 6. Quality histogram for the remeshing of a human pelvis. Comparison for the harmonic mapping and the convex combination mapping.

for every element T_k. However, the parametrization resulting from a convex combination map is much more distorted than the standard harmonic one. The equivalent PDE resulting from the convex combination map is an anisotropic diffusion problem, with a piecewise constant diffusion tensor that is equal to $\mathbf{M}_\xi/\sqrt{\det \mathbf{M}_\xi}$. Surfaces with highly distorded metrics are known to make the work of surface meshers more difficult: Figure 6 compares meshes of a human pelvis generated using either standard harmonic mappings or convex combination maps. The quality of elements clearly deteriorates when using convex combination maps. This can be explained by the simpler example of Figure 7. Here, we start from a very bad triangulation (Figure 7(a)). We parametrize it using both harmonic and convex combination maps. Iso-values of the x-coordinate are drawn for both maps on the unit disk. Even though the mesh issued from the convex combination map is much smoother in the parametric plane than the standard harmonic one, isovalues are much closer to straight lines for the standard map. This means that a straight line in the parameter plane is close to a straight line in the real plane for the harmonic map and hence that the metric tensor $\mathbf{M}$ (8) is much smoother for the harmonic map than it is for the convex combination map.

To solve the problems associated with convex combination maps, we propose a more local way to enforce discrete one-to-one map. The algorithm is called *cavity check* (see Figure 8) and goes as follows:

1. Compute harmonic map using finite elements;
2. For every interior vertex V_i of the parameter plane, check if each of its neighboring triangles (defining a cavity) is oriented properly[‡];
3. If elements are reversed, move the vertex at the center of gravity of the kernel of the polygon P surrounding the point (see Figure 8).

To find the kernel of a star-shaped polygon, different algorithms have been proposed in the literature [30, 31]. In this work, as the polygons have a small number of vertices we have implemented a simple quadratic algorithm. It should be noted that the situation presented in Figure 8(a) does occur very rarely and only occurs for very poor quality initial stl files (one or two cases at most for the examples presented in Section 5).

[‡]This is simply implemented by comparing normal orientations of the triangles in the parametric space.

DOI: 10.1002/nme

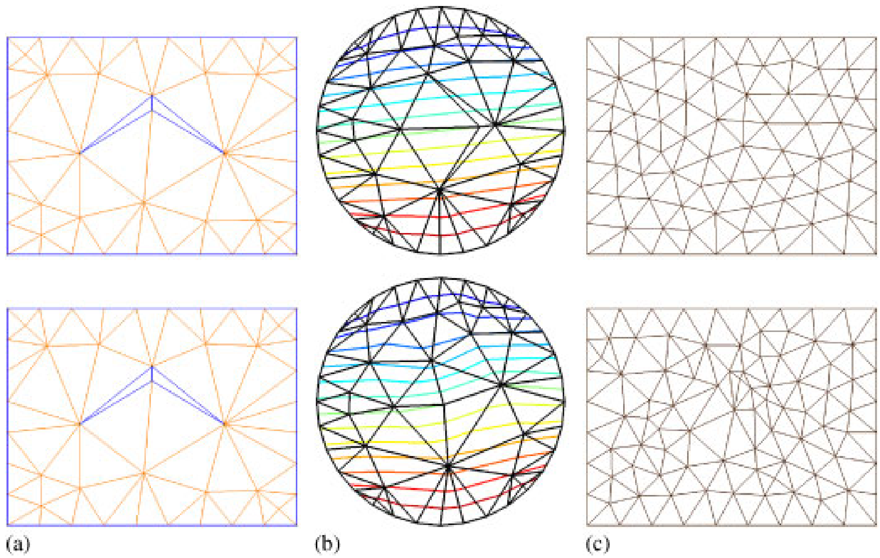


Figure 7. Poor quality initial triangulation (a) that has been remeshed using a harmonic map (top figures) and a convex combination map; (bottom figures): (b) mapping of the initial mesh onto the unit disk with iso-x values; and (c) the final mesh. For this example a direct mesher based on local mesh modifications (Gmsh/meshadapt) is used to remesh the parametrized surface.

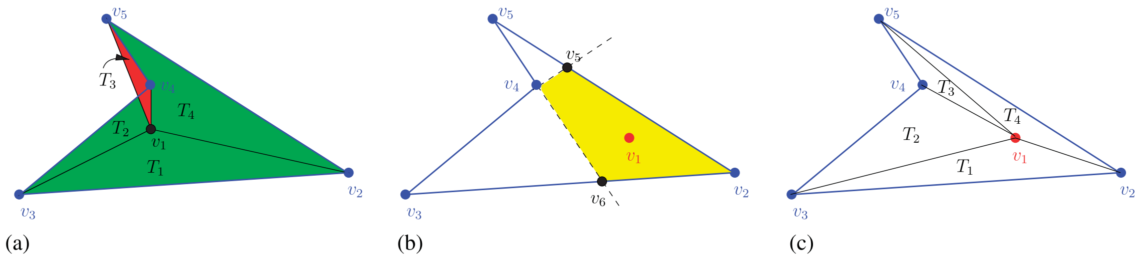


Figure 8. Vertex v_1 with four neighboring triangles $T_1=(v_1, v_2, v_3)$, $T_2=(v_1, v_3, v_4)$, $T_3=(v_1, v_4, v_5)$, $T_4=(v_1, v_5, v_2)$ defining the polygon $P=(v_2, v_3, v_4, v_5)$. (a) Triangle T_3 is not well oriented and overlaps the triangles T_2 and T_4; (b) The vertex v_1 is moved and placed inside the kernel of the polygon P; and (c) Now, all four triangles become well oriented without overlapping and hence the discrete mapping is guaranteed to be one-to-one for this cavity.

3.3. Harmonic maps for geometries with large aspect ratio

In some cases for which the ratio between the equivalent diameter of the closed loop $\partial\mathscr{S}_1$ and the length in the direction normal to the surface with line loop $\partial\mathscr{S}_1$ is too high, we fail to compute the harmonic map.

In order to explain this, we take a simple example of a surface $\mathscr{S}$ that is a cylinder of height H and radius R. We can easily compute analytically the harmonic map on this cylinder. Indeed,

DOI: 10.1002/nme

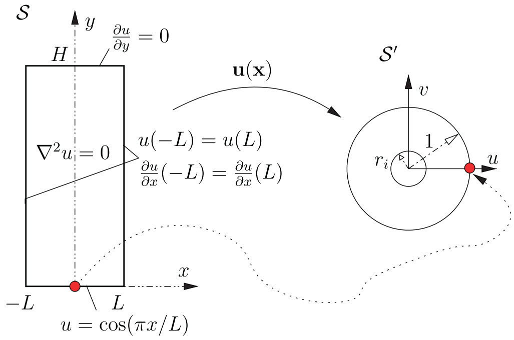


Figure 9. Harmonic mapping of the cylinder onto the unit disk. The left figure shows the rectangular domain of size $[2L \times H]$ and the boundary conditions used to compute the analytical solution of the Laplace equation on the cylinder of height H and radius $R = L/\pi$. The right figure shows the harmonic map on the unit disk.

finding the harmonic map is equivalent to solving the two Laplace Equations (11) on a rectangle of height H and length $2L$, with $L = \pi R$ with the boundary conditions shown in Figure 9. The analytical solution of the problem is

$$u(x, y) = \cos\left(\frac{\pi x}{L}\right) Y(y), \quad v(x, y) = \sin\left(\frac{\pi x}{L}\right) Y(y) \tag{20}$$

where

$$Y(y) = \cosh\left(\frac{\pi y}{L}\right) - \left(\tanh\left(\frac{\pi H}{L}\right) \sinh\left(\frac{\pi y}{L}\right)\right). \tag{21}$$

The function $Y(y)$ rapidly tends to zero. This means that, for high geometrical ratios ($H/R \approx 6\pi$), computed coordinates become non distinguishable for high y's because of the computer finite precision. Figure 10 shows the map of the cylinder into an annulus. The inner radius of the annulus goes to zero exponentially. In practice, using double precision arithmetic we have experienced that Gmsh's meshers will fail to deliver decent meshes when $H/R > 6\pi$. The algorithm can be improved by scaling the problem by the geometrical aspect ratio of the cylinder H/R and by solving the following Laplace equation with anisotropic coefficients k_x and k_y:

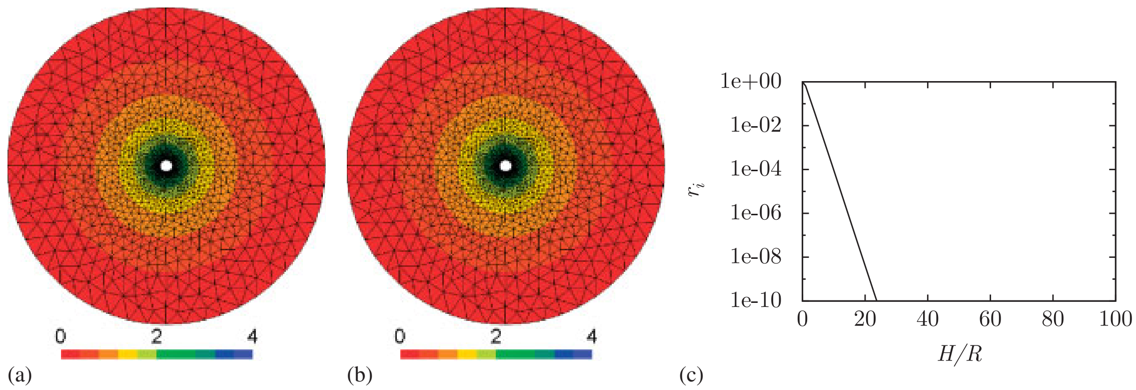
$$k_x u_{,xx} + k_y u_{,yy} = 0, \quad k_x v_{,xx} + k_y v_{,yy} = 0, \quad \text{with } k_x = 1, k_y = H/R. \tag{22}$$

The solution in the y-direction is then given by:

$$Y(y) = \frac{e^{\frac{-y}{\sqrt{H}\sqrt{R}}} \left(e^{\frac{2y}{\sqrt{H}\sqrt{R}}} + e^{\frac{2\sqrt{H}}{\sqrt{R}}}\right)}{e^{\frac{2\sqrt{H}}{\sqrt{R}}} + 1}. \tag{23}$$

As can be seen in Figure 11 this function is less stiff and the inner radius r_i does not tend to zero for large values of the geometrical aspect ratio.

 Int. J. Numer. Meth. Engng 2010; **83**:403–425
DOI: 10.1002/nme



(a) (b) (c)

Figure 10. Harmonic map of the cylinder of height H and radius R onto the unit disk. Mesh in the unit circle and values of $y(u, v)$ for (a) $H/R=2$ ($r_i=0.26$); (b) $H/R=4$ ($r_i=0.036$); and (c) $r_i=\sqrt{u(L,H)^2+v(L,H)^2}$ as a function of H/R.

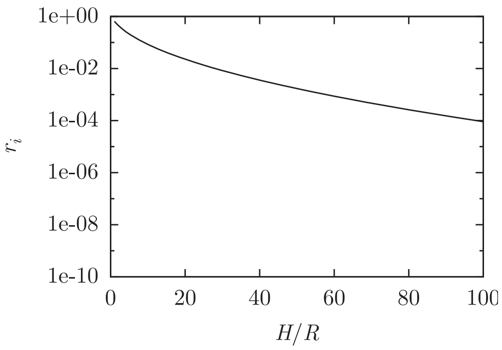


Figure 11. $r_i=\sqrt{u(L,H)^2+v(L,H)^2}$ as a function of H/R for the scaled Laplacian problem (22).

This technique could be generalized in different ways. The first idea is to simply compute global anisotropic coefficients from the size of the oriented bounding boxes [32]. A second idea is to compute local anisotropic coefficients from the gradient of the distance function to the boundary $\partial\mathscr{S}_1$. The distance function could be for example computed as the solution of an elliptic PDE as described in [33]. Another possibility, usually used in computer graphics, is to use a partition scheme based on the concept of Voronoi diagrams [21] or inspired by Morse theory [18, 34], which would lead to a partitioning of the mesh into a number of charts that by construction have a uniform geometrical aspect ratio.

3.4. Higher order representation of the geometry

In case of faceted triangulations, it is often desirable to smooth the normals when remeshing. This may be the case for example when the CAD is described by very few triangles or for triangulations obtained from crude segmentation techniques.

This can be achieved by changing only one of the three maps $\mathbf{x}(\xi)$ defined in Figure 2. Instead of choosing linear finite elements for this mapping as is done in (3), we have chosen in this work

DOI: 10.1002/nme

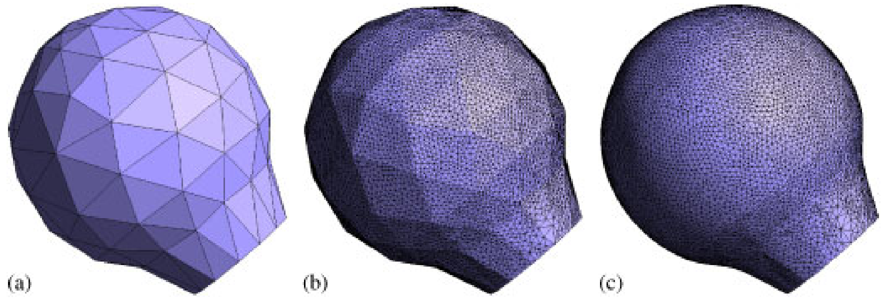


Figure 12. (a) Initial STL triangulation; (b) remeshing with a linear map $\mathbf{x}(\xi)$; and (c) remeshing with a cubic map $\mathbf{x}(\xi)$.

a cubic interpolation that is often used in the community of computer graphics [35, 36]:

$$\begin{aligned}\mathbf{x}(\xi) = {} & a_{300}\zeta + a_{030}\xi + a_{003}\eta + a_{210}3\zeta^2\xi + a_{120}3\zeta\xi^2 + a_{201}3\zeta^2\eta \\ & + a_{021}3\xi^2\eta + a_{102}3\zeta\eta^2 + a_{012}3\xi\eta^2 + a_{111}6\xi\eta\zeta, \quad \text{with } \zeta = 1-\xi-\eta. \end{aligned} \tag{24}$$

The coefficients a_{ijk} are the 9 control points of the curved PN triangle [36] that can be computed from the triangle $T_j \in \mathscr{R}^3$ that is defined by its three coordinates $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ and its three vertex normals $\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{n}_3$:

$$\begin{aligned} a_{300} &= \mathbf{x}_1, \quad a_{030} = \mathbf{x}_2, \quad a_{003} = \mathbf{x}_3, \\ w_{ij} &= (\mathbf{x}_j - \mathbf{x}_i)\cdot \mathbf{n}_i, \\ a_{210} &= (2\mathbf{x}_1 + \mathbf{x}_2 - w_{12}\mathbf{n}_1)/3, \quad a_{120} = (2\mathbf{x}_2 + \mathbf{x}_1 - w_{21}\mathbf{n}_2)/3, \\ a_{021} &= (2\mathbf{x}_2 + \mathbf{x}_3 - w_{23}\mathbf{n}_2)/3, \quad a_{012} = (2\mathbf{x}_3 + \mathbf{x}_2 - w_{32}\mathbf{n}_3)/3, \\ a_{102} &= (2\mathbf{x}_3 + \mathbf{x}_1 - w_{31}\mathbf{n}_3)/3, \quad a_{201} = (2\mathbf{x}_1 + \mathbf{x}_3 - w_{13}\mathbf{n}_3)/3, \\ E &= (a_{210} + a_{120} + a_{021} + a_{012} + a_{102} + a_{201})/6, \quad V = (\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3)/3, \\ a_{111} &= E + (E - V)/2. \end{aligned} \tag{25}$$

The advantage of the curved PN triangles is that they are very straightforward to implement and provide a smoother, though not necessarily everywhere tangent, continuous surface [36]. Figure 12 shows an initial triangulation and the new meshes computed with harmonic maps for both a linear and cubic mapping $\times(\xi)$. As can be seen, the cubic mapping enables to smooth nicely the initial faceted triangulation.

The advantage of this method compared to smoothing-based approaches (used in the context of direct methods) is that our technique does not result in any feature loss.

4. COMPUTATIONAL ALGORITHM

The presented algorithm for remeshing consists of different steps as illustrated in Figure 13:

1. Start from an initial triangulation.
2. Compute the mapping:
   (a) Divide the surface into surfaces of genus $G = 0$.

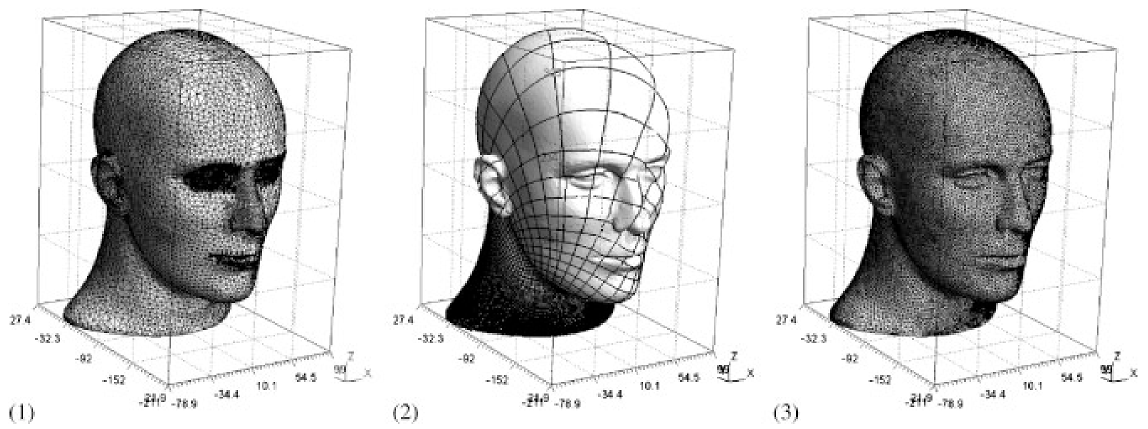


Figure 13. Remeshing algorithm. (1) Initial triangulation, (2) Harmonic map $u(\mathbf{x})$ and $v(\mathbf{x})$, and (3) new mesh based on the harmonic map.

   (b) Solve two Laplace equations for computing u and v using finite elements.
   (c) Verify that the discrete harmonic map is locally one-to-one. If not, proceed as explained in Section 3.2.
3. Use standard surface meshers to remesh in the parametric space and map the triangulation back to the original surface.
4. From the surface mesh, use standard volume meshers to build a 3D finite element mesh.

We will now detail some of the steps involved in the remeshing algorithm cited above.

For step 2(a), as explained in Section 2, the surface should be of zero genus and have at least one boundary. In case the conditions to compute the harmonic map are not satisfied, the mesh is split into different parts that each satisfy the conditions. We are currently working on an optimal automatic splitting algorithm (numerical homology) that will be presented in an upcoming paper. For step 2(b), we use the high-performance direct solver TAUCS to solve the linear system that arises from the finite element discretization of the Laplace Equation (16). For step (3), we mesh in the parametric space such that all edges $\mathbf{e}$ have a non-dimensional length of $l_e = 1$, where the non-dimensional length is defined as:

$$l_e = \int_e \frac{1}{\delta(\mathbf{x})}\,\mathrm{d}l, \tag{26}$$

with δ denoting the mesh size field [25] and where dl is given by (9).

5. EXAMPLES

The parametrization and remeshing procedure described above has been implemented in the open source finite element mesh generator Gmsh [25]. We present several examples for which this approach may be of interest, namely STL triangulations and triangulations that come from a CAD representation. We then show results of direct mesh generation based on surfaces that are parametrized with harmonic maps.

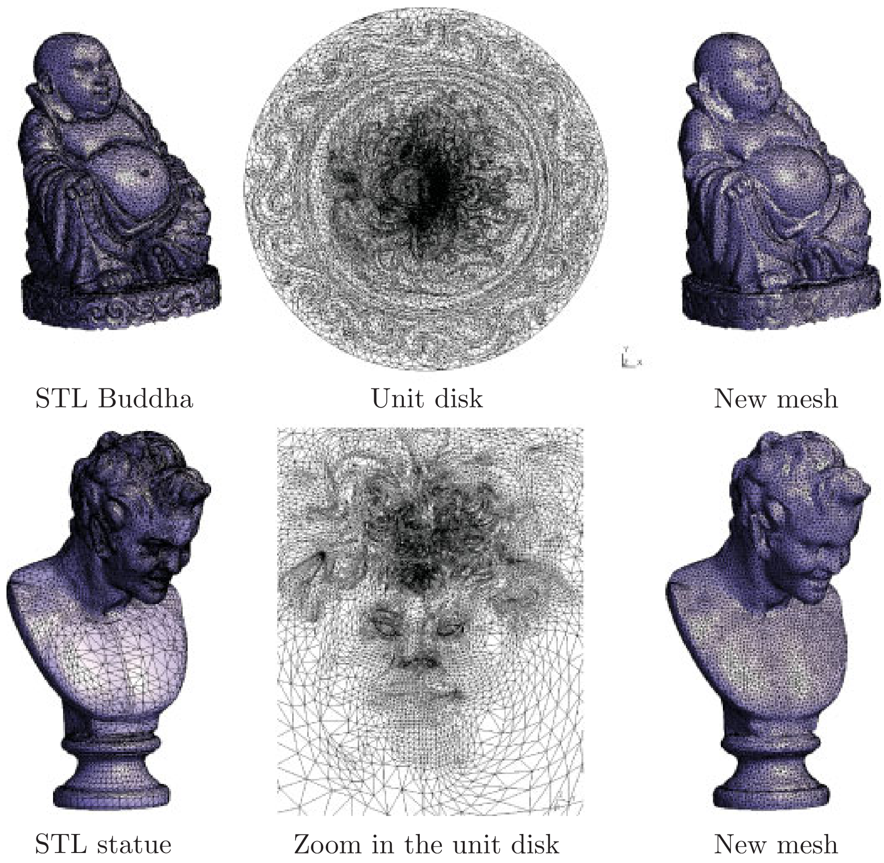


Figure 14. Two examples of parametrizations of STL triangulations.

5.1. Remeshing (low-input) STL triangulations

In this section, we show examples of surfaces that do not have a standard CAD representation such as STL triangulations. Those triangulations can be found in many domains such as 3D scanned images, computer game characters, terrain data, and medical data issued from a segmentation. Figure 14 shows some examples of parametrizations of triangulated surfaces. The examples were found on the INRIA web site[§] of Eric Saltel.

Figure 15 shows the quality histogram for the initial STL triangulation of a pelvis and arterial bypass presented in Figure 3 and the remeshed geometry. The quality histogram shows the aspect ratio of the surface mesh that is defined as:

$$\eta = K \frac{\text{inscribed radius}}{\text{circumscribed radius}} \tag{27}$$

where K has been chosen so that the equilateral triangle has $\eta = 1$.

[§] `http://www-c.inria.fr/Eric.Saltel/saltel.php`.

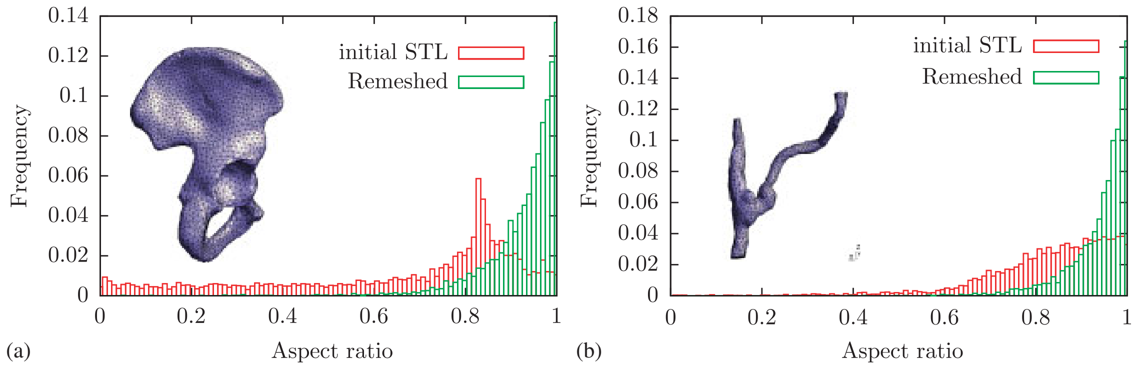


Figure 15. Plot of the quality histogram of both the STL triangulation and the remeshed part of (a) a pelvis and (b) a bypass of a femoral artery.

As the surface of the pelvis is of genus $G=1$, one cut is made[¶] to generate two surfaces of zero genus and a parametrization based on harmonic maps is performed for each of those two new genus zero surfaces. We can see that with the remeshing procedure, we greatly enhance the quality of the mesh.

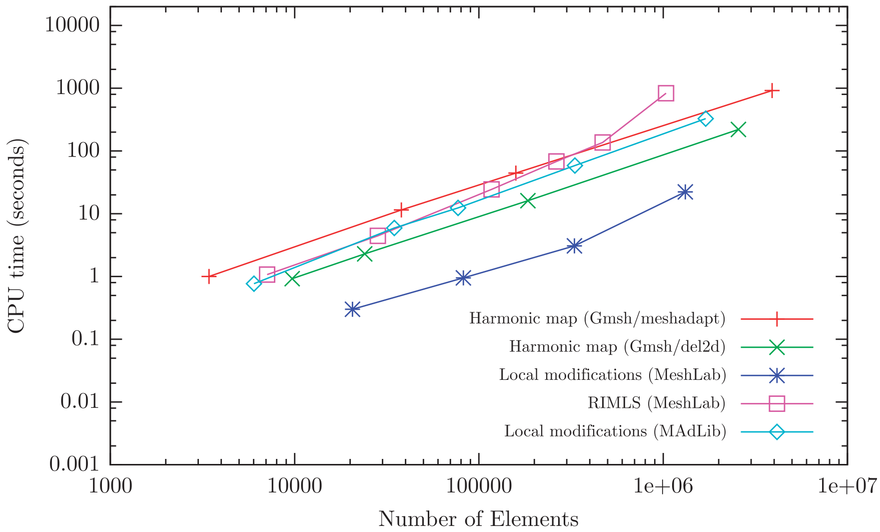
Figure 16 compares the efficiency and quality of the proposed method with three alternative remeshing algorithms:

- Local mesh modifications in MeshLab [37] combines a planar flipping optimization algorithm with a subdivision surfaces method [40] that aims at smoothing the surface by successive refinements of the mesh. In order to establish a comparison with the other methods, we apply a zero threshold angle, which leads to a uniform refinement over the mesh.
- The Robust Implicit Moving Least Squares (RIMLS) method described in [39] and implemented in MeshLab [37], whose goal is to obtain smooth representations of surfaces while preserving fine details. We notice that this method is designed for graphical purposes rather than computational meshes.
- The local mesh modification strategy described in [41] and implemented in the MAdLib package [38, 42], which modifies the initial mesh to make it comply with criteria on edge lengths and element shapes by applying a set of standard mesh modifications (edge splits, edge collapses and edge swaps, ...) in an optimal order.

For the purpose of comparison only uniform size fields are prescribed in this test. The computations were performed on a MacBook Pro 2.33 GHz Intel Core 2 Duo. We can see that the additional computational effort is quite reasonable for our method and is worth it compared to direct remeshing methods since the mean quality of the mesh is about $\bar{\eta}=0.94$ for the harmonic map (Gmsh/del2d) method and respectively $\bar{\eta}=0.78$ for the direct methods that use local mesh modifications with MAdLib and $\bar{\eta}=0.85$ with MeshLab. Only the subdivision method is faster than the other methods, but it is not designed to handle 3D meshes or prescribed element sizes since the refinement level depends only on the angle between the triangles. Another observation is that our method behaves well regarding the low-quality elements. Finally, the RIMLS method gives meshes with

[¶]The cut can be easily realized with Gmsh by using the Mesh > Reclassify tool.

DOI: 10.1002/nme



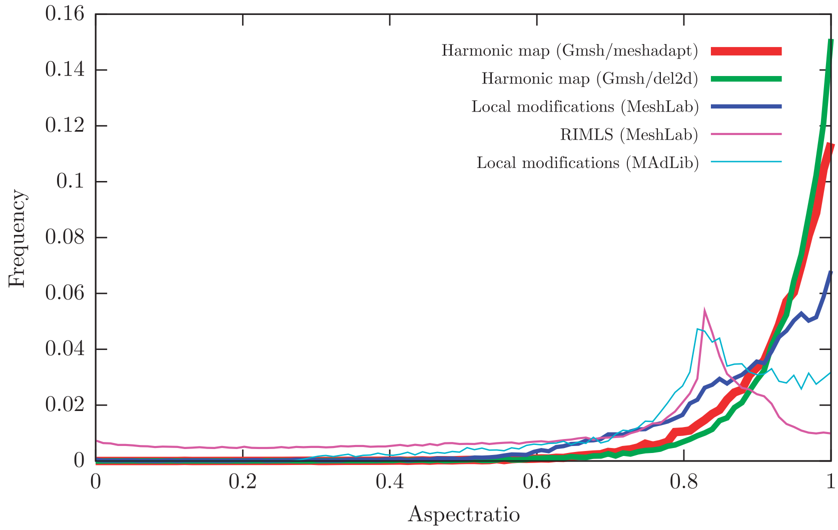


Figure 16. Comparison of the proposed method based on harmonic maps (using two different surface meshing algorithms: Gmsh/meshadapt and Gmsh/del2d [25]) with two direct remeshing methods based on local mesh modifications (MeshLab [37] and MAdLib [38]) and with the RIMLS reparametrization method [39]. Top: comparison of the CPU time requirement. Bottom: comparison of the quality of the surface meshes. (The quality is found to be independent of the mesh size.)

both low quality elements and a poor mean quality ($\bar{\eta}=0.65$) since it is not intended to produce computational meshes.

Figure 17 shows surface and volume meshes created from low-quality triangulations for actual biomedical applications: an arterial bifurcation and a pelvis. Figure 17(b) shows how the remeshed surface can be used to construct high-quality boundary layer meshes for cardiovascular blood flow simulations using MAdLib [38, 42].

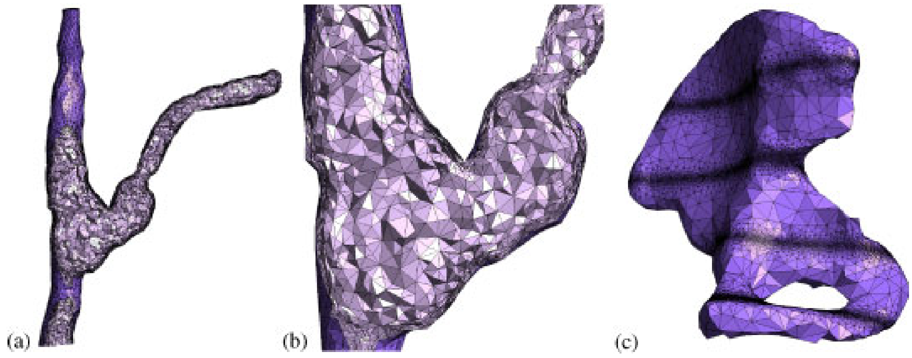


Figure 17. Meshes created from STL triangulations obtained from CT-scans: (a) Arterial bifurcation with a uniform edge length on the boundary and a boundary layer mesh; (b) zoom of the boundary layer; (c) pelvis with a sinusoidal edge length.

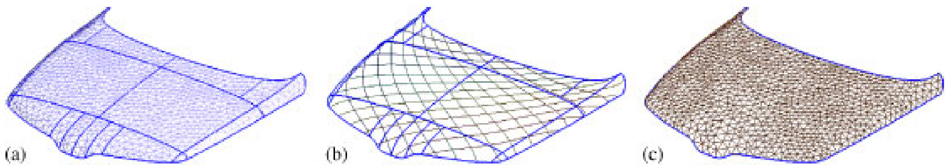


Figure 18. Surface mesh improvement with a single grouped patch and a parametrization with a harmonic mapping: (a) Meshed CAD model with multiple patches; (b) inverse parametrization $\mathbf{u}(\mathbf{x})$ of all the patches with a single harmonic map; and (c) new mesh of the compound.

5.2. Remeshing CAD patches

The next example shows a CAD model of a car hood made of nine different patches that are smoothly connected together (Figure 18). Standard surface meshers mesh each of those patches separately as shown in Figure 18(a). One common issue in engineering analysis is the presence of small sub-patches for the description of one smooth surface, which induces the presence of small elements, leading to difficulties in the finite element analysis. It is therefore highly useful to reparametrize those patches into one single surface. This has been done in two steps: a mesh has been generated on the multiple patches (Figure 18(a)) and the reparametrization (Figure 18(b)) has been computed on this first mesh. Then, we can remesh the whole compound using any of the surface mesh generators available (Figure 18(c)). Note that, in the case of multiple CAD patches reparametrization, points on the reparametrized surface are subsequently projected on the exact CAD model. We use an initial triangulation that is conforming to the patches so that every triangle of this initial triangulation lies on only one CAD patch. It is then easy to compute local CAD coordinates of every point in the new mesh and to compute thereafter their exact location on the CAD model.

The example of Figure 19 shows another CAD model composed of multiple patches that has been reparametrized into three patches. This example shows clearly the advantage of the approach when coarse meshes have to be generated. Here, reparametrizing a geometry

DOI: 10.1002/nme

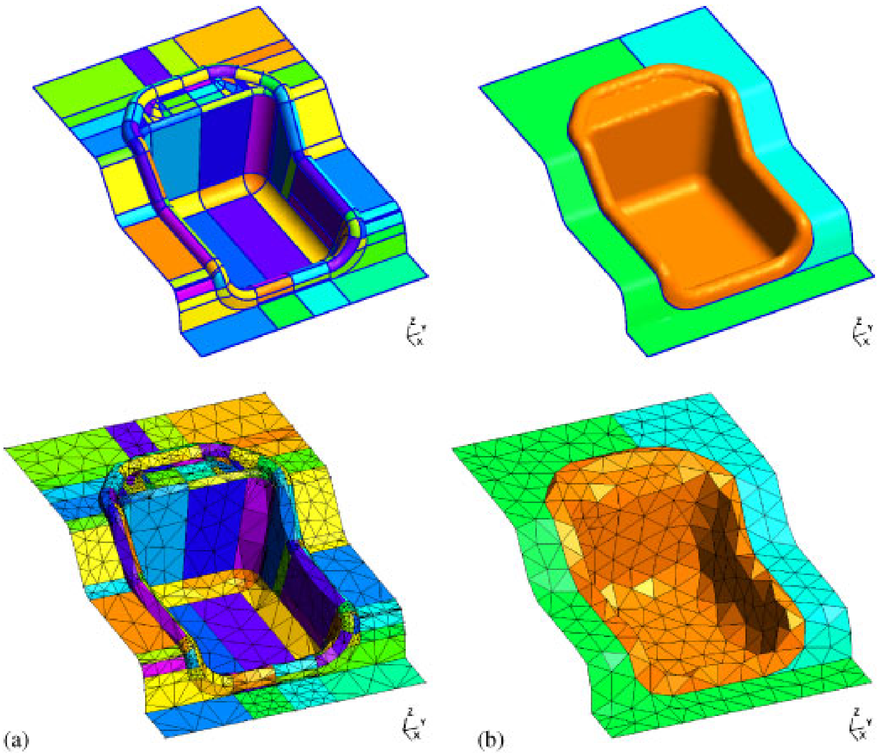


Figure 19. Meshing CAD surfaces composed of multitudes of Bezier patches (courtesy of SAMTECH): (a) Meshing all the patches separately and (b) Using harmonic maps to reparametrize and remesh the CAD into only three patches.

allows to generate smooth uniform meshes. Small details in the initial CAD model induce the generation of small ill-shaped elements. Note the number of reparametrized surfaces was done arbitrarily.

As another an example of a moderately complicated CAD model, we consider the Airbus A319. The aircraft is initially composed of 89 surface patches. After reparametrization, this number has been reduced to 25 (Figure 20). Moreover, lots of curves were reparametrized, especially at the junction between the wings and the fuselage. Figure 20 shows both initial and reparametrized CAD models. Figure 21 shows a global view of the surface mesh as well as a zoom on the mesh of the left wing. The mesh size was adapted to the curvature of the model. Figure 22 shows a large part of the fuselage that has been reparametrized. The reparametrized patch has three holes: two for the wings and one for the right end part. Those three holes, which are clearly visible in the parametric plane, are highly distorted. Even though, Gmsh's surface meshers were able to produce high-quality meshes (Figure 21) with such highly distorded input.

5.3. Remeshing in Gmsh

As previously mentioned, the remeshing algorithm based on harmonic maps is implemented within the open-source software Gmsh. We show a simple example of how to use it. We suppose that we

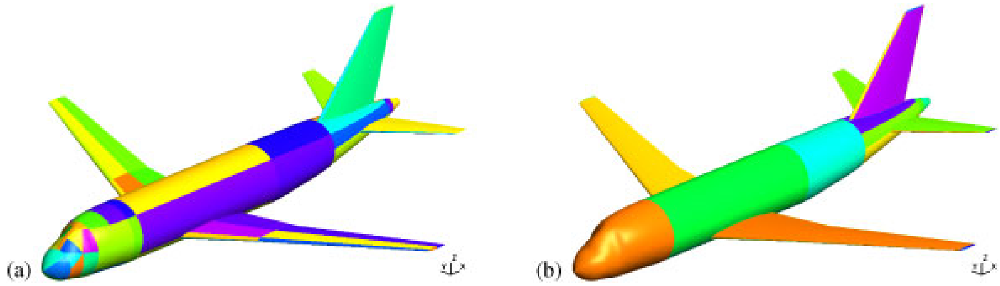


Figure 20. The model of the airbus A319: (a) The initial CAD data made of 89 patches and (b) reparametrized CAD made of only 25 patches.

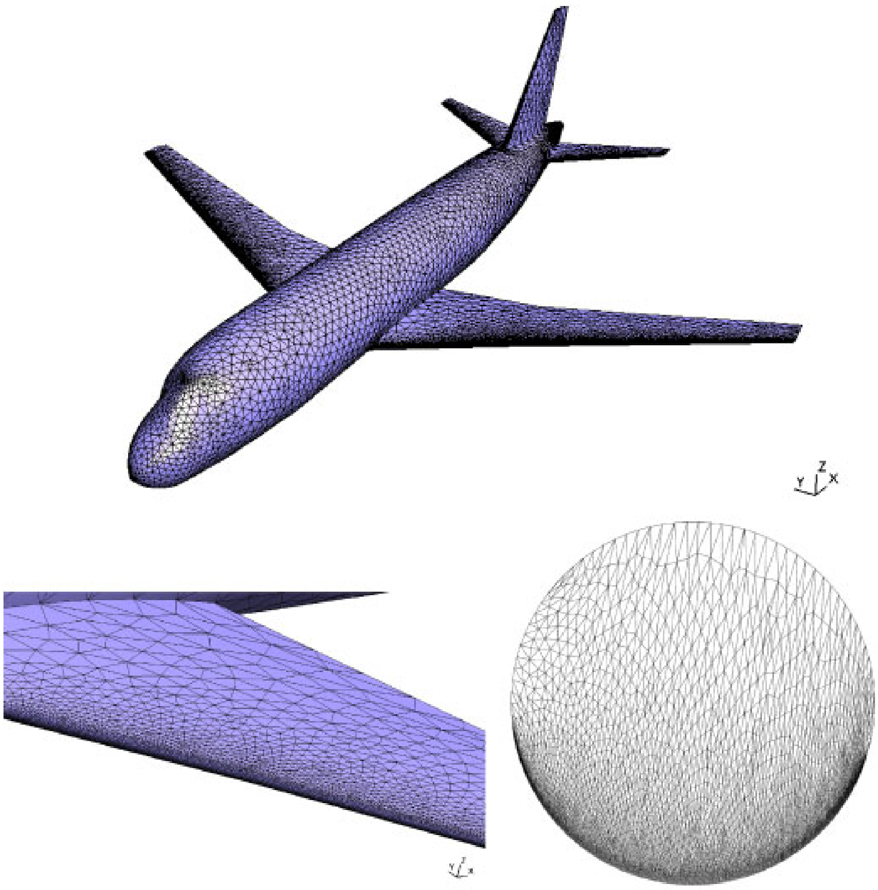


Figure 21. Figure shows the surface mesh of the airbus A319 for which the mesh size is adapted to the principal curvature of the model. Top figure shows a global view of the mesh, whereas bottom figures show a zoom of the mesh of a wing, both in the real space and in the unit circle.

DOI: 10.1002/nme

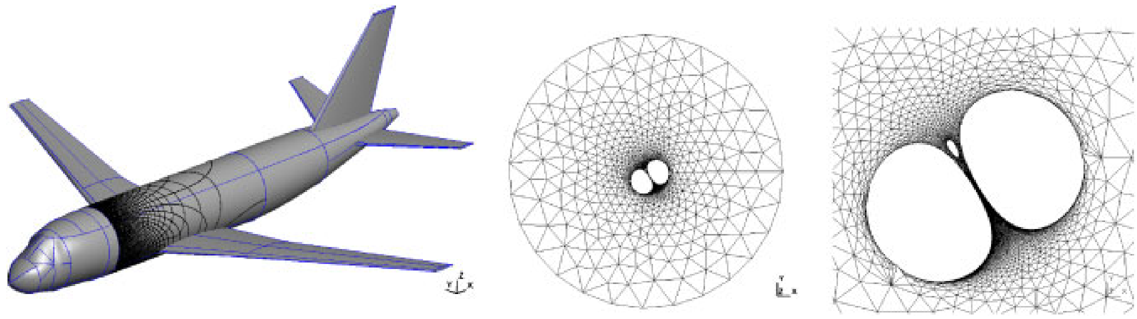


Figure 22. Reparametrization of a large part of the fuselage. Left figure shows iso-contours of u and v. Center and right figures show the mesh in the parametric plane, i.e. inside the unit disk.

have an initial surface mesh and write the following text file ‘remesh.geo’:

```
// Merge initial mesh (in .stl, .msh, .mesh, .brep, .medit, etc. format)
Merge "bypass.stl";

// If the initial mesh contains different topological entities,
// then re-create the topology
CreateTopology;

// If necessary, create a topological volume
Surface Loop(55) = {15, 16};
Volume(56) = {55};

// Remesh the Edges and Faces (and Volumes) with harmonic maps
Compound Line(10) = {2, 3}; // merge 2 edges
Compound Surface(100) = {1:24}; // auto-detect boundary
Compound Volume(1000) = {56}
```

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented an efficient method for surface and subsequent volume remeshing. The method is based on the parametrization of a genus zero surface with a harmonic map.

We have presented the discrete finite element harmonic map with appropriate boundary conditions. We have introduced a local ‘cavity check’ algorithm to enforce the discrete one-to-one mapping and showed that this approach leads to higher quality meshes than the convex combination map of Floater. A higher order approximation of the geometry based on curved PN triangles was introduced to smooth faceted STL triangulations. Our procedure is easy to implement and very robust against low-quality input triangulations. Compared to smoothing-based approaches our technique does not result in any feature loss and naturally offers refinement options (boundary layer mesh, curvature, etc.). As it enables to remesh multiple CAD patches, the approach can be used to substantially reduce the time required to prepare CAD surface definition for surface mesh generation. The time required to generate the surface mesh is less than $100s$ per 10^6 elements.

DOI: 10.1002/nme

Furthermore, the generated elements have a high mean quality measure, which is a clear demonstration of the suitability of the meshes for finite element simulations.

We are currently working on an optimal numerical homology algorithm that will automatically cut a initial surface onto different surfaces of genus zero with uniform geometrical aspect ratio. With this upcoming algorithm we hope to obtain a fully automatic method for high-quality remeshing of any topological surface without any geometrical constraint.

ACKNOWLEDGEMENTS

J.-F. Remacle and C. Geuzaine thank Dr. B. Levy for the fruitful discussions they had on reparametrization techniques at the Trophées du Libre 2009 in Soissons.

REFERENCES

1. Shewchuk JR. What is a good linear element? Interpolation, conditioning, and quality measures. In *11th International Meshing Roundtable*, Sandia National Laboratories, Ithaca, New York, 2002; 115–126.
2. Szczerba D, McGregor R, Szekely G. High quality surface mesh generation for multi-physics bio-medical simulations. *Computational Science—ICCS 2007*, vol. 4487. Springer: Berlin, 2007; 906–913.
3. Marcum DL, Gaither A. Unstructured surface grid generation using global mapping and physical space approximation. *Proceedings of the 8th International Meshing Roundtable*, South Lake Tahoe, CA, 1999; 397–406.
4. Aftosmis M, Delanaye M, Haimes R. Automatic generation of cfd-ready surface triangulation from cad geometry. *AIAA Paper* 1999; **1**:09-0776.
5. Ito Y, Nakahashi K. Direct surface triangulation using stereolithography data. *AIAA Journal* 2002; **40**(3):490–496.
6. Bechet E, Cuilliere JC, Trochu F. Generation of a finite element mesh from stereolithography (stl) files. *Computer-Aided Design* 2002; **34**(1):1–17.
7. Wang D, Hassan O, Morgan K, Weatheril N. Enhanced remeshing from stl files with applications to surface grid generation. *Communications in Numerical Methods in Engineering* 2007; **23**:227–239.
8. Borouchaki H, Laug P, George P. Parametric surface meshing using a combined advancing-front generalized delaunay approach. *International Journal for Numerical Methods in Engineering* 2000; **49**:223–259.
9. Zheng Y, Weatherill N, Hassan O. Topology abstraction of surface models for three-dimensional grid generation. *Engineering Computations* 2001; **17**:28–38.
10. Marcum DL. Efficient generation of high-quality unstructured surface and volume grids. *Engineering Computations* 2001; **17**:211–233.
11. Tristano J, Owen S, Canann S. Advancing front surface mesh generation in parametric space using riemannian surface definition. *Proceedings of 7th International Meshing Roundtable*. Sandia National Laboratory: Dearborn, MI, 1998; 429–455.
12. Laug P, Boruchaki H. Interpolating and meshing 3d surface grids. *International Journal for Numerical Methods in Engineering* 2003; **58**:209–225.
13. Attene M, Falcidieno MSB, Wyvill G. A mapping-independent primitive for the triangulation of parametric surfaces. *Graphical Models* 2003; **65**:260–273.
14. Bennis C, Vézien JM, Iglésias G. Piecewise surface flattening for non-distorted texture mapping. *ACM SIGGRAPH Computer Graphics*, Las Vegas, NV, 1991; 237–246.
15. Maillot J, Yahia H, Verroust A. Interactive texture mapping. *Proceedings of ACM SIGGRAPH'93*, Anaheim, 1993; 27–34.
16. Floater MS. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design* 1997; **14**:231–250.
17. Greiner G, Hormann K. Interpolating and approximating scattered 3d data with hierarchical tensor product splines. In *Surface Fitting and Multiresolution Methods*, Le Mehaute A, Rabut C, Schumaker LL (eds). Chamonix, France, 1996; 163–172.
18. Levy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. *Computer Graphics* (*Proceedings of SIGGRAPH 02*), San Antonio, TX, 2002; 362–371.
19. Sheffer A, Praun E, Rose K. Mesh parameterization methods and their applications. *Foundations and Trends in Computer Graphics and Vision* 2006; **2**(2):105–171.

20. Floater MS, Hormann K. Surface parameterization: a tutorial and survey. *Advances in Multiresolution for Geometric Modelling*. Springer: Berlin, Heidelberg, 2005.
21. Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W. Multiresolution analysis of arbitrary meshes. *SIGGRAPH '95*: *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, Los Angeles, CA, 1995; 173–182.
22. Schoen R, Yau S. *Lectures on Harmonic Maps*. International Press Harvard University: 1997.
23. Alliez P, Meyer M, Desbrun M. Interactive geometry remeshing. *Computer Graphics* (*Proceedings of the SIGGRAPH 02*), San Antonio, CA 2002; 347–354.
24. Floater MS. Parametric tilings and scattered data approximation. *International Journal of Shape Modeling* 1998; **4**:165–182.
25. Geuzaine C, Remacle JF. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering* 2009; **79**(11):1309–1331.
26. Rado T. Aufgabe 41. *Math-Verien*, 1926; 49.
27. Choquet C. Sur un type de représentation analytique généralisant la représentation conforme et défininie au moyen de fonctions harmoniques. *Bulletin Des Sciences Mathematiques* 1945; **69**:156–165.
28. Floater MS. One-to-one piecewise linear mappings over trinagulations. *Mathematics of Computation* 2003; **72**:685–696.
29. Pal M, Edwards MG. Quasi-monotonic continuous darcy-flux approximations in 3-d for any element type. *SPE Reservoir Simulation Symposium*, Houston, TX, U.S.A., 2007.
30. Lee DT, Preparata F. An optimal algorithm for finding the kernel of a polygon. *Journal of the ACM* 1979; **26**(3):415–421.
31. Icking C. Earching for the kernel of a polygon: a competitive strategy using self-approaching curves. *11th Annual ACM Symposium on Computational Geometry*, Vancouver, 1995; 258–266.
32. Chang CT, Gorissen B, Melchior S. Fast computation of the minimal oriented bounding box on the rotation group so(3). *ACM Transactions on Graphics*, 2009; submitted.
33. Legrand SED, Hanert E, Legat V, Wolanski E. High-resolution unstructured meshes for hydrodynamic models of the great barrier reef, Australia. *Estuarine*, *Coastal and Shelf Science* 2006; **68**(1–2):26–46.
34. Shinagawa Y, Kunii T, Kergosien YL. Surface coding based on morse theory. *IEEE Computer Graphics and Applications* 1991; **11**(5):66–78.
35. Boubekeur T, Alexa M. Phong tessellation. *ACM Transactions on Graphics* 2008; **27**(5):141:1–141:5.
36. Vlachos A, Peters J, Boyd C, Mitchell JL. Curved pn triangles. *Proceedings on 2001 Symposium on INteractive 3D Graphics*, Research Triangle Park, NC, 2001; 159–166.
37. 3D-CoForm. Meshlab 2009. http://meshlab.sourceforge.net/.
38. Compère G, Remacle JF. A mesh adaptation framework for large deformations. *International Journal for Numerical Methods in Engineering* 2009; accepted.
39. Oztireli A, Guennebaud G, Gross M. Feature preserving point set surfaces based on non-linear kernel regression. *EUROGRAPHICS* 2009; **28**(2).
40. Loop C. Smooth subdivision surfaces based on triangles. *Master's Thesis*, Department of Mathematics, University of Utah, August 1987.
41. Compère G, Remacle JF, Marchandise E. Transient mesh adaptivity with large rigid-body displacements. In *Proceedings of the 17th International Meshing Roundtable*, vol. 3, Garimella R (ed.). Springer: Berlin, 2008; 213–230. ISBN: 978-3-540-87920-6.
42. Compère G, Remacle JF. Website of madlib: mesh adaptation library 2008. http://www.madlib.be.

DOI: 10.1002/nme

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2010; **00**:1–6 *Prepared using* nmeauth.cls *[Version: 2002/09/18 v2.02]*

High Quality Surface Remeshing Using Harmonic Maps. Part II: Surfaces with High Genus and of Large Aspect Ratio.

E. Marchandise[1], C. Carton de Wiart [1,4], W. G. Vos[3], C. Geuzaine[2] and J-F Remacle[1*]

[1] *Université catholique de Louvain, Institute of Mechanics, Materials and Civil Engineering (iMMC), Place du Levant 1, 1348 Louvain-la-Neuve, Belgium*
[2] *Université de Liège, Department of Electrical Engineering and Computer Science, Montefiore Institute B28, Grande Traverse 10, 4000 Liège, Belgium*
[3] *University Hospital Antwerp, Department of Pulmonology , Antwerp, Belgium*
[4] *Cenaero, Rue des Frères Wright 29, 6041 Gosselies, Belgium*

SUMMARY

This paper follows a previous one that was dealing with high quality surface remeshing using harmonic maps [1]. In [1], it has been demonstrated that harmonic parametrizations can be used as input for surface meshers to produce high quality triangulations. However, two important limitations were pointed out, namely surfaces with high genus and/or of large aspect ratio. This papers addresses those two issues. We first develop a multiscale version of the harmonic parametrization of [1] and then combine it with a multilevel partitioning algorithm to come up with an automatic remeshing algorithm that overcomes the above mentioned limitations of harmonic maps. The overall procedure is implemented in the open-source mesh generator Gmsh [2]. Copyright © 2010 John Wiley & Sons, Ltd.

KEY WORDS: surface remeshing; surface parametrization; STL file format; surface mapping; harmonic map; conformal map; finite elements

1. Introduction

Most of the surface meshing procedures require a parametrization of the surface. The surface mesh is generated in the parametric plane and is subsequently projected onto the 3D surface using the parametrization. Yet, there are situations when the only description of the surface is a triangulation. In the latter case, the geometric triangulation can usually not be used for a finite element analysis. One can either modify the geometric triangulation in 3D or use the triangulation for building a discrete parametrization.

The main driving force in the research on discrete parametrization techniques is Computer Graphics (CG) where discrete parametrizations are used among other things for texture mapping. Various approaches have been proposed in the CG literature and it is possible to

*Correspondence to: emilie.marchandise@uclouvain.be

Received
Revised

classify them as follows: linear, non-linear and hybrid methods. Linear algorithms are efficient, yet they usually do not guarantee the discrete mapping to be one-to-one. It is possible to combine linear algorithms with local checks to guarantee its bijectivity: we called that approach hybrid [1]. Non-linear algorithms are considered to be slow and will not be considered here.

The concept of discrete harmonic map has been described first by Eck [3]. Floater et al. [4] introduced a concept of convex combination map that guarantees that the discrete mapping is one-to-one. Sheffer and Strurler [5] presented a constrained minimization approach, the so-called angle based flattening (ABF), such that the variation between the set of angles of an original mesh and one of the 2D flattened version is minimized. In order to obtain a valid and flipping-free parametrization, several additional constrained algorithms are developed. More recently, they improved the performance of the ABF technique by using an advanced numerical approach and a hierarchical technique [6]. Much research has also been incorporated within the theory of differential geometry. For example, Levy et al [7] apply the Cauchy Riemann equations to compute a least square conformal map. This approach is quite similar to that of Desbrun [8] that minimize a combination of Dirichlet and distortion energy to compute a conformal map for interactive geometry remeshing. More recently, Mullen et al. [9] have also presented spectral computation of the conformal map [9].

Recently, discrete parametrizations methods have received some attention from non CG-specialists and in particular, in the domain of finite element mesh generation. Here, the target application is clearly surface meshing, especially for surfaces that are defined by a triangulation only or for cross-patch meshing. In [1], we have demonstrated that the use of such mappings as parametrizations allowed to generate quality finite element meshes. Yet, two important issues have not been completely addressed: reparametrization techniques fail when the surface has a large aspect ratio and/or when it has a high genus. Those issues are critical in the domain of mesh generation, maybe more than in computer graphics. This paper aims at addressing those two issues.

In [1], we have demonstrated that parametric coordinates computed using a discrete harmonic map become exponentially small for vertices located away from the boundaries of the surface to be remeshed. This makes any 2D remeshing procedure fail because the local coordinates become numerically undistinguishable. If this issue has already been tackled by Alliez et al. in the CG community [10], their proposed solution procedure can however not be used in the context of surface remeshing. In a few words, the idea of Alliez et al. is to partition the zero genus mesh of large geometrical aspect ratio into two parts. The partition is defined by first computing an harmonic map and by evaluating the area distortion map that indicates how the triangles have been shrunk or expanded during the parametrization. Then a vertical line is drawn in the parametric space that passes through the center of gravity of the distortion map and that defines the partition of the mesh. If the authors in [10] could partition this way some simple surfaces with moderate aspect ratio, we found that this area-distortion partitioning algorithm fails for larger geometrical aspect ratio for the same reason the remeshing procedure fails (numerically indistinguishable coordinates). Here, we thus extend the idea of Alliez and develop the concept of multiscale harmonic maps.

The second issue that we address deals with the parametrization of surfaces that are not homeomorphic to a unit disk, i.e., surfaces that do not have zero genus with at least one boundary. In the CG community, many authors use a partition scheme based on the concept of Voronoï diagrams [3] or inspired by Morse theory [7, 11]. The resulting mesh partitions are area-balanced patches that are disk-like. However, this approach results in a large number of patches and hence a large number of interfaces between those patches. A large number of patches is however not desirable in the context of remeshing because it constrains the final mesh to have mesh edges on those interfaces. Other CG authors introduce seam generation techniques [7, 12] that generate cuts in the surface that are positioned in areas where they cause no texture artifacts. More recently, some authors suggested different methods to compute parametrizations that are globally smooth with singularities occurring at only a few extraordinary vertices [13, 14]. Even though the latter two techniques are attractive in the context of texture mapping, they are less efficient in terms of computational time than partitioning methods combined with a local parametrization method.

The solution strategies presented in this paper for the remeshing of surfaces with large geometrical aspect ratio and arbitrary genus are independent of the chosen type of linear harmonic maps. Among the harmonic maps, we have implemented the Laplacian harmonic map onto a unit disk [1], the convex combination map onto a unit disk [1, 4] and a finite element least square conformal map with open boundaries [9, 10]. The different linear harmonic maps along with the solution strategies presented in this paper are implemented in the open source mesh generator software Gmsh [2]. The different examples show that our method is of high interest for remeshing biomedical triangulations that have a very poor quality input triangulation, or for industrial CAD-based surfaces that contain too many tiny surfaces that are not appropriate for finite element computations.

2. Parametrization of a mesh with harmonic maps

Parametrizing a surface $\mathcal{S}$ is defining a map $\mathbf{u}(\mathbf{x})$

$$\mathbf{x} \in \mathcal{S} \subset \mathcal{R}^3 \mapsto \mathbf{u}(\mathbf{x}) \in \mathcal{S}^* \subset \mathcal{R}^2 \tag{1}$$

that transforms continuously a 3D surface $\mathcal{S}$ into a surface $\mathcal{S}^*$ embedded in $\mathcal{R}^2$ that has a well known parametrization. Such a continuous parametrization exists if the two surfaces $\mathcal{S}$ and $\mathcal{S}^*$ have the same topology, that is have the same genus $G(\mathcal{S})$ and the same number of boundaries $\partial\mathcal{S}_i$, $i = 1, ..., N_B$. The genus $G(\mathcal{S})$ of a surface is the number of handles in the surface[†].

In this work, we consider that the only available representation of a surface $\mathcal{S}$ is a conforming triangular mesh $\mathcal{T}$ in 3D, i.e. the union of a set of triangles that intersect only at common vertices or edges. Let us consider a triangulated surface that has n_V vertices, n_E edges and n_T triangles. The genus $G(\mathcal{T})$ is then given through the Euler-Poincaré formula:

$$G(\mathcal{T}) = \frac{-n_V + n_E - n_T + 2 - N_B}{2}, \tag{2}$$

where N_B is the number of boundaries of the triangulation.

As stated in the introduction, harmonic maps have been chosen for the parametrization because they are easy to compute. As an example, the discrete Laplacian harmonic map can

[†]For example, a sphere has a genus $G = 0$ and $N_B = 0$, a disk has $G = 0$ but $N_B = 1$ and a torus has $G = 1$ and $N_B = 0$

be computed as follows:

$$\begin{cases} \Delta_\epsilon u = 0, & \Delta_\epsilon v = 0 \quad \text{in} \quad \mathcal{T}, \\ u = u_D, & v = v_D \quad \text{on} \quad \partial\mathcal{T}_1, \\ \partial_n u = 0, & \partial_n v = 0 \quad \text{on} \quad \partial\mathcal{T}\backslash\partial\mathcal{T}_1 \end{cases}, \tag{3}$$

where Δ_ϵ denotes the discrete Laplacian operator than can be easily computed with piecewise linear finite elements, and $\mathbf{u}_D(\mathbf{x})$ is the value for the Dirichlet boundary conditions. Figure 1 shows for example such a parametrization for a triangulation of a cylinder. The triangulation has two boundaries $N_B = 2$ and the lowest one is mapped onto the circle of the unit disk by imposing $u_D(\mathbf{x}) = \cos(2\pi l(\mathbf{x})/L)$ and $v_D(\mathbf{x}) = \sin(2\pi l(\mathbf{x})/L)$, with l denoting the curvilinear abscissa of a point along the boundary $\partial\mathcal{T}_1$ of total length L (red arrow in Fig. 1).

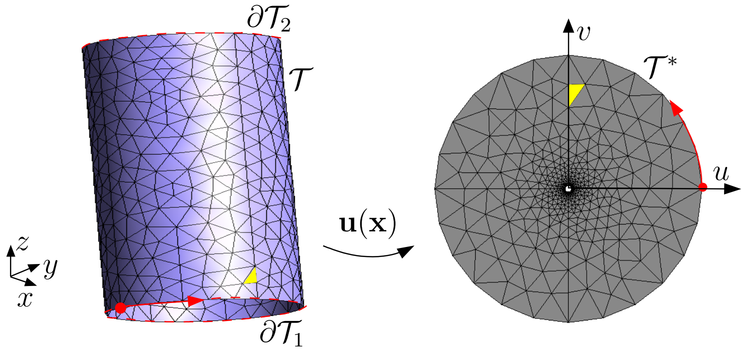


Figure 1. Parametrization. A piecewise linear map creates a correspondence between a 3D surface mesh $\mathcal{T}$and a 2D mesh $\mathcal{T}^*$ of same topology $(G = 0, N_B = 2, \eta = 2)$, mapping each triangle from $\mathcal{R}^3$ to $\mathcal{R}^2$.

When going from continuous harmonic maps to linear discrete harmonic maps (3), three different issues can arise. The first issue concerns undistinguishable mapping coordinates. As shown in Fig.1 and as explained in [1] the solution of the mapping becomes exponentially small for vertices located away from $\partial\mathcal{T}_1$. As a consequence, local coordinates u and v of those distant vertices might numerically become indistinguishable. By deriving an analytical solution of Laplacian harmonic maps for cylinders, we showed in [1] that the geometrical aspect ratio of the surface η should be reasonable to be numerically able to distinguish the coordinates.

$$\eta = \frac{H}{D} < \eta_{\max}, \tag{4}$$

where H is the maximal geodesic distance between a mesh vertex and a boundary vertex of $\partial\mathcal{T}_1$ and D is the equivalent diameter of the boundary $\partial\mathcal{T}_1$. As the distance on the 3d mesh is not straightforward to compute, an upper and lower bound for η is computed. The upper bound for η is computed by using the analytical expression for cylinders: $\eta = 2\pi A/L^2_{\partial\mathcal{T}_1}$, where A is the area of the 3D surface mesh and $L_{\partial\mathcal{T}_1}$ is the arc length of the boundary $\partial\mathcal{T}_1$. The lower bound is estimated by choosing for H the maximal size of the oriented bounding box of the 3D surface mesh and for D the maximal size of the oriented bounding box for the boundary curve $\partial\mathcal{T}_1$. The oriented bounding boxes are computed with the fast Oriented bounding box HYBBRID optimization algorithm presented in [15] which combines the genetic and Nelder-Mead algorithms [16].

As we showed in [1] that $\eta = 4$ corresponds to an area of mapped triangles of about $r_i^2 = 10^{-10}$ (see Eq. (23) and Fig. 10c in [1]), we choose $\eta_{\max} = 4$ as upper limit for the geometrical aspect ratio of the 3D surface mesh. A second issue is about triangle flipping. As the discrete harmonic map has no guarantee to be one-to-one, we suggested in [1] a *local cavity check algorithm* that locally modifies mesh cavities in which flipping occurs. In the case the algorithm fails, we suggest to switch to a guaranteed one-to-one convex combination mapping introduced by Floater [17, 18]. This convex combination is not used as default mapping since the metric tensor associated with this mapping is much more distorted than the one obtained with the harmonic mapping and hence the resulting new mapped mesh is of lower quality. Finally, the last issue concerns triangle flipping that might occur when computing linear conformal maps with open boundaries. Indeed, a linear algorithm cannot guarantee that no triangle folding will occur. To remain efficient in the context of surface remeshing, we use an idea similar to the one suggested by Sheffer et al. [5] that checks the presence of edge folding.

In this section, we have put to the fore in the context of discrete harmonic mapping three limitations, namely limitations on the genus G, the number of boundaries N_B and the geometrical aspect ratio η. Those three criteria can be summarized as follows:

$$\begin{aligned} &i) \quad G = 0;\\ &ii) \quad N_B \geq 1;\\ &iii) \quad \eta < \eta_{\max}. \end{aligned} \tag{5}$$

In the next section, we will first consider a disk-like surface ($G = 0, N_B \geq 1$) and present a novel max-cut partitioning algorithm for those family of surfaces. We will then present an automatic approach for arbitrary genus surfaces that combines this novel algorithm with a multilevel partitioning algorithm.

3. Multiscale Laplace partitioning method

The multiscale algorithm that we present here is an extension a method proposed in [10] that aims at building area-balanced maps. Recall briefly the idea of the max-cut partitioning algorithm of Alliez and his co-authors. Consider the cylindrical surface of Figure 1. In the parametric plane (u, v), elements far from the boundary $\partial\mathcal{T}_1$ have areas that rapidly tend to zero: the map is not area-balanced. The method proposed in [10] is to build a split line in the parametric plane. If we assume that every vertex $\mathbf{u}_i(u_i, v_i),\ i = 1, \ldots, N$ of the triangulation has a unit weight, it is possible to compute both the center of gravity $\mathbf{u}^c = \sum_i \mathbf{u}_i/N$ and the axes of inertia of the set of vertices as the eigenvectors of

$$\mathbf{J} = \left[\begin{array}{cc} \sum_i (u_i^c - u_i)^2 & \sum_i (u_i^c - u_i)(v_i^c - v_i) \\ \sum_i (u_i^c - u_i)(v_i^c - v_i) & \sum_i (v_i^c - v_i)^2 \end{array} \right].$$

The split line considered by Alliez et al. passes through the center of gravity and has the direction of the principal axis of inertia. It splits the surface in two parts and produces a quality partitioning of the surface. This "max-cut" partitioning algorithm enables to partition a non-closed triangulated surface mesh that has a large geometrical aspect ratio (think of arteries, bunny ears, lungs) into two parts. The two mesh partitions can then be subsequently remeshed by computing two different parametrizations. Yet, the method relies on the fact that it is possible to classify triangles in 2D with respect to the partition line. When the aspect ratio is too high, the nodes that are located far away from the Dirichlet boundary have coordinates that cannot be distinguished: it may be impossible to determine if a node is either on the left or on the right of the partition line. Moreover, because of the finite precision of linear system solvers, the mesh in the parameter space may become invalid.

Consider a cylinder of height H and diameter D with an aspect ratio $\eta = H/D = 25$. Applying Alliez's method with a geometry with that kind of aspect ratio leads to the generation of a mesh that is incorrect in the parameter space. Figure 2 shows the result of the harmonic map parametrization into the unit circle. Triangles get smaller and smaller when they get close to the center of the circle. Figure 2 shows all triangles that have areas smaller than 10^{-15}. Those small triangles correspond roughly to 30% of the total amount of triangles of the mesh. The center of gravity $\mathbf{u}^c$ is situated too high so that the split does not partition the mesh in an optimal way, as it is depicted on Figure 2.

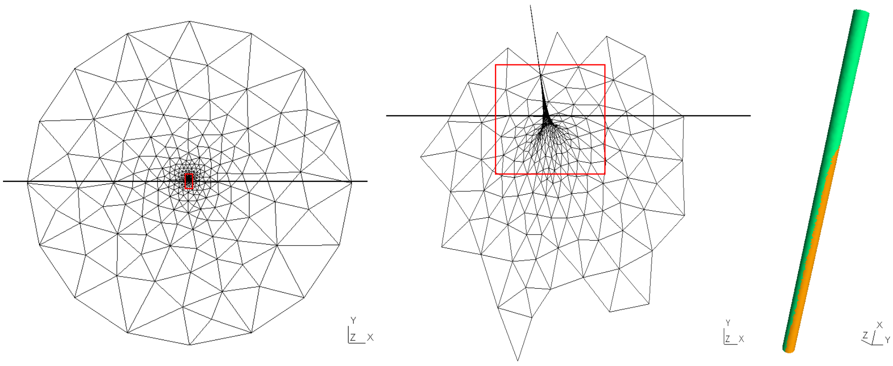


Figure 2. Partitioning a cylinder of aspect ratio $\eta = 25$. Left figure shows the mesh in the parameter space with the computed split line. Center figure shows a zoom in the parameter space with all triangles of size smaller than 10^{-15}. Right figure shows the (incorrect) partitioned mesh that relies on the split line.

The behavior of Alliez's method can become erratic for more complex geometries. Consider the example of Figure 3. Here we consider the geometry of an aorta with an aspect ratio $\eta \simeq 17$[‡]. The surface has $N_B = 13$ boundaries. The image of the partition line is far from being smooth in the real space and the resulting partition is not usable.

Here, we propose a multiscale partitioning algorithm that is a generalization of the partitioning algorithm of Alliez et al. Consider a surface with a high aspect ratio. The elements of the parametric plane that have too small areas form m clustered regions of the plane. Figure

[‡]the geometry has been downloaded from the simbios web site https://simtk.org/frs/download.php?file_id=662

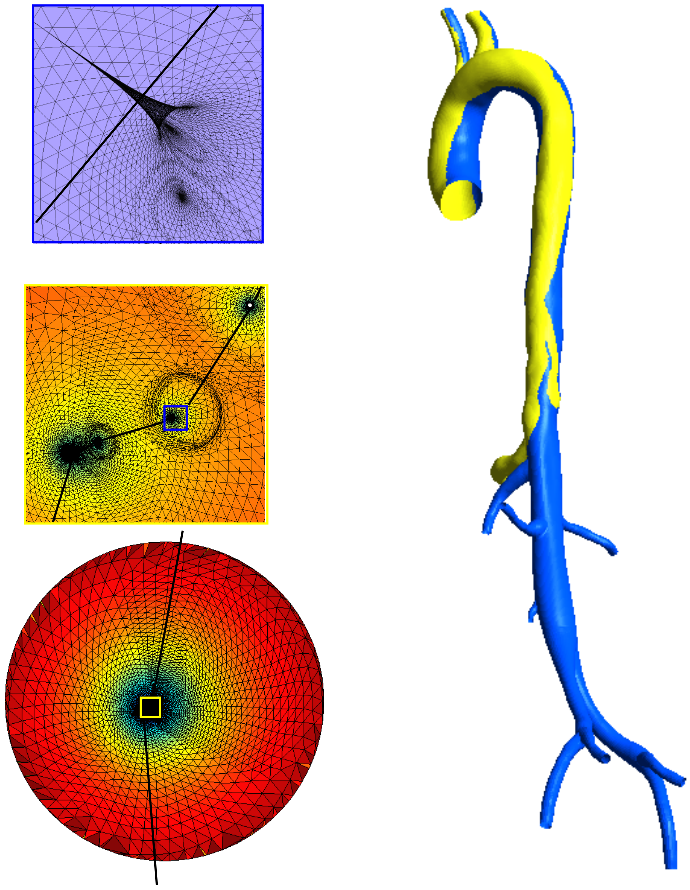


Figure 3. Partitioning an aorta of aspect ratio $\eta = 17$ with $N_B = 13$ boundaries using the area balanced partitioning method of Alliez. Top left figure shows the mesh in the parametric space. The color indicates the area distortion map, that is, the ratio between the area of each triangle in the 2D space and the triangles area in the 3D space. The black line shows the partition line that should split the mesh into two area balanced mesh partitions. However as can be seen on the left middle figure, the mapping could not be computed correctly due to numerical round-offs errors and triangles have been flipped. The resulting mesh partition is presented in the right figure.

4 shows an illustration of the first step of the multiscale partitioning algorithm for a mesh of an aortic artery.

Each of those m subregions can be translated to its center of gravity and rescaled so that its bounding box is of size 1. A new map per subregion can therefore be computed. The latter procedure can be applied recursively to those m subregions until every element of the map has a computable (i.e. sufficiently large) area. At the end, a split line is defined through all levels of the recursion, leading to a procedure that splits the surface into two parts that can

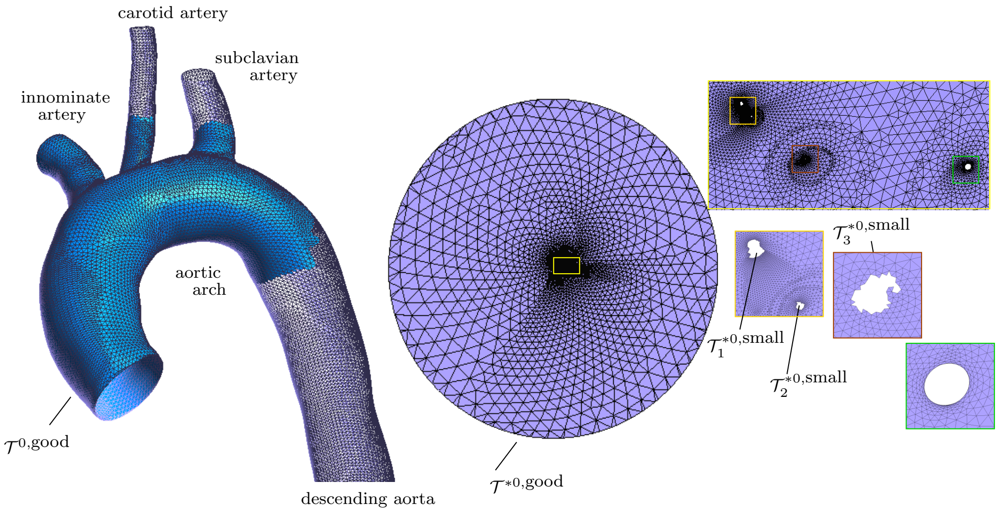


Figure 4. First step of the multiscale partitioning algorithm for the geometry of an aorta. The triangles that have a too small area in the parametric plane define here two clustered regions. Those regions are then withdrawn from the parametrization. The three small white regions visible in the parametric space $\mathcal{T}_i^{0,\text{small}}$, $i = 1, ..., 3$ correspond respectively in the 3D space to the subclavian artery, the carotid artery, and the descending aorta.

be easily parametrized i.e. for which no element is far from the unique boundary.

Our multiscale partitioning method is based on the computation for n levels of discrete Laplacian harmonic maps (denoted $\mathbf{u}^{ij}(\mathbf{x})$, $i = 0, ..., n$) and reads as follows (see Fig. 5):

1. At Level 0. Parametrize the surface triangulation $\mathcal{T}$ with a Laplace harmonic map $\mathbf{u}^{00}(\mathbf{x})$ and a Dirichlet boundary condition that maps one of the boundaries of the mesh $\partial\mathcal{T}_1$ onto the unit circle (red line on Fig.5). The parametrization is computed as the solution of the PDE's equations (3) that are discretized with linear finite elements on the initial STL triangulation (see paper [1] for more details about the harmonic mapping computation).

2. The triangles that have an acceptable size are tagged in the parametric space as $\mathcal{T}^{*0,\text{good}}$ (corresponding to $\mathcal{T}^{0,\text{good}}$ in the real space). See for example the blue part of the geometry on Fig. 5. The triangles that have a mapped area smaller than $\mathcal{A}_{2D}^{\text{min}} = 1.e^{-10}$ are withdrawn from level 0 and are put in sets of connected small triangles $\mathcal{T}_j^{*0,\text{small}}, j = 1, ..., m$. In the case of Fig. 5, there are three sets $(m = 3)$ of connected small triangles for level 0[§].

3. At Level $1j$. For each connected triangulation $\mathcal{T}_j^{0,\text{small}}$, define the parametrization $\mathbf{u}^{1j}(\mathbf{x})$ with a Laplace harmonic map and a Dirichlet boundary condition that maps the

[§]Note that the pathological sets of triangles which contain only very few triangles are not taken into account. The triangles belonging to those sets are then tagged as $\mathcal{T}^{*0,\text{good}}$.

boundary $\mathcal{T}^{0,\text{good}} \cap \mathcal{T}_j^{0,\text{small}}$ onto the polygon defined in the 2D parametric space at level 0: $\mathbf{u}_D = \alpha \mathbf{u}^0(\mathbf{x})$. Note that this polygon is scaled with a parameter α that is such that the equivalent diameter for $\mathcal{T}_j^{*0,\text{small}}$ has a value of 1.

4. For the levels $i = 2, ..., n$, repeat step 2 and 3 by replacing level 0 and 1 respectively by level $i-1$ and i until the set of small elements at level i ($\mathcal{T}_j^{*i,\text{small}}$) is empty.

5. Define in the parametric space a splitting line that connects recursively the center of gravities of the small mapped connected triangles $\mathcal{T}_j^{*i,\text{small}}$ to the left and right of the unit disk. For the last level, we take either the barycenter of the elements $\mathcal{T}_j^{*i,\text{good}}$ or the centers of the closed loops inside the 2D mapping (see for example mapping 11 and 30 in Fig. 5).

It must be noted that the multiscale procedure defined here somehow aims at computing one single harmonic mapping with a machine precision that exceeds the double precision. Some of the sub-problems consist in mapping 3D sub-triangulations onto non convex regions: this, in principle, does not guarantee those sub-mappings to be one-to-one. Yet, boundary conditions of the sub-problems arise from upper recursion levels, in a way that the global multiscale problem aims at solving one single harmonic map from $\mathcal{S}$ onto a unit circle (which is convex). This proves that every sub-problem will be well posed and will provide sub-mappings that are one-to-one.

This new multiscale partitioning method has allowed us to partition rather complex surfaces, with very high aspect ratios and a large number of boundaries. Figure 6 presents two examples of the multiscale partitioning algorithm applied to biomedical geometries. A correct partitioning of the aorta of Figure 5 is presented.

We have also run our algorithm for a more complex case: the geometry of human airways of very large geometrical aspect ratio $\eta = 89$ (Figure 6, right). Segmentation of the lung models is performed on low dose CT images, voxel size of $0.5\,mm$, using the commercially available Mimics software package (Materialise, Belgium). A semi-automatic segmentation algorithm places all identifiable airway branches in a separate mask. The smooth 3D models are generated from the segmentation masks. These models include upper airways, all central airways, and the distal airways with a minimal diameter of $1-2\,mm$. This corresponds to terminal airways in the 5th-9th generation. The smoothed models are cut perpendicular to the airway centerline to provide well defined inlet and outlet surfaces. The resulting STL triangulation of the lung models satisfies the topological conditions (5) (i)-(ii)($G=0, N_B=147$) so that our multiscale approach can be applied. In what follows, a recursion depth of $n=15$ has often been reached.

4. Automatic remeshing

In this section, a fast and automatic algorithm is developed that overcomes the three topological and geometrical limitations of harmonic maps (see conditions (5)).

Topological conditions (i) and (ii) require the surface to have a genus $G=0$ and at least one boundary ($N_B \geq 1$). The way to go is to split the surface into partitions that have the right topology. If $G>0$, then it can be shown that it is possible to define G independent cuts that divide the surface into subsurfaces that all have $G=0$. There exist specialized *computational*

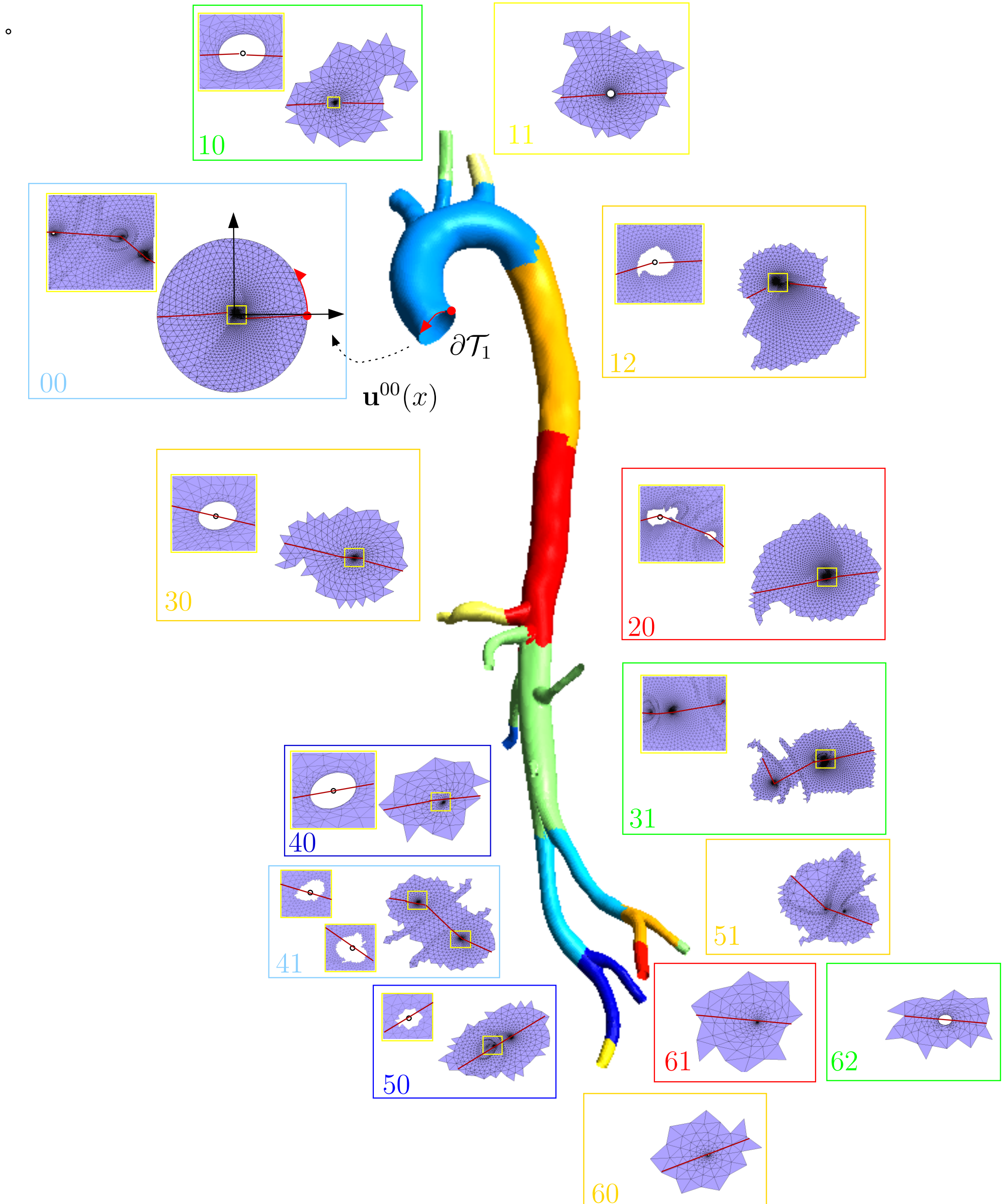


Figure 5. Multiscale Laplace partitioning method of an aorta($G = 0, N_B = 13, \eta = 17$). In this example there are $n = 6$ different levels on which harmonic maps are computed. The red line shows the partition line that recursively splits the mesh into two area balanced mesh partitions (see the resulting mesh partition in Fig.6).

Prepared using nmeauth.cls

a) b)

Figure 6. Examples of the multiscale Laplace partitioning method applied to geometries with large geometrical aspect ratio η: a) aorta ($\eta = 17$) and b) human airways ($\eta = 89$).

homology algorithms that aim at constructing such a minimum set of cuts [19]. Yet, those algorithm are usually slow (they act on a coarsened version of the triangulation, with the same topology and use a greedy algorithm to determine the cuts) and they do not always produce optimal cuts in term of their "shape". Here, we look at simpler solutions, even if more than G cuts are used. We use a multilevel edge partitioning software such as Chaco [20] or Metis [21] and partition recursively the triangulation into a minimal number of partitions that satisfy the topological and geometrical conditions. For the geometrical condition (iii) to be satisfied, we use the partitioning method based on the above presented multiscale harmonic map.

The automatic procedure for remeshing a triangulation $\mathcal{T}$ is illustrated in Figure 7 and reads as follows:

1. First, check conditions (i)-(ii) by computing the genus of the triangulation using (2) and the number of boundaries N_B. If the topological conditions are not satisfied, recursively split the mesh with the multilevel partitioning method until satisfied.

2. Next, check condition (iii) by computing approximately the geometrical aspect ratio η from the analytical expression for cylinders of height H and diameter D: $\eta = H/D \approx \pi A/L^2$, where A is the area of the triangulation and L is taken to be the maximal curvilinear length of all the closed boundaries of the triangulation ($L = 2\pi R$ for the

Prepared using nmeauth.cls

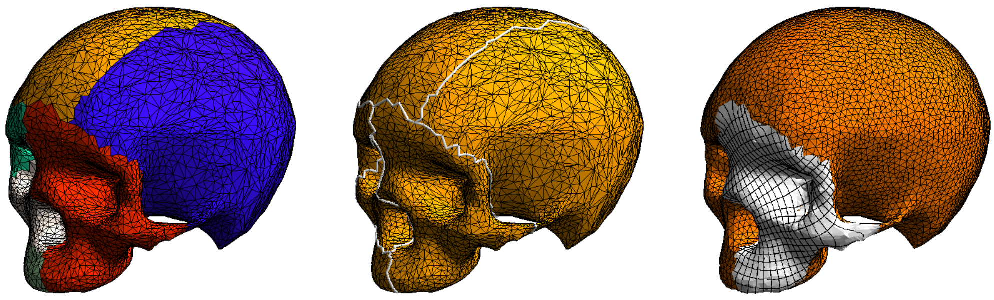


Figure 7. Remeshing algorithm for a skull geometry. a) Initial triangulation ($G = 2, N_B = 0$) that is cut into different mesh partitions of zero genus, b) Remesh the lines at the interfaces between partitions, c) Compute a harmonic map for every partition and remesh the partition in the parametric space ($\mathbf{u}(\mathbf{x})$ coordinates visible for one partition).

cylinder). In case this condition is not satisfied, split the mesh into two parts with the multiscale laplacian partitioner defined in the previous section.

3. Compute an armonic mapping for every mesh partition.

4. Remesh the lines that are the boundaries of the triangulation and the interfaces between the mesh partitions (see the interfaces between colored patches in 7a that are marked with thick white lines in Fig.7b). Those lines are defined as model edges and divided into N parts as follows: $N = \int_0^L ||\mathbf{x}_{,t}|| /\delta dt$, where δ is the prescribed mesh size field. The remeshed lines are embedded in the final mesh (see Fig. 7c).

5. Use standard surface meshers to remesh every partition in the parametric space and map the triangulation back to the original surface.

6. If a volume mesh is needed, generate a volume mesh from the new surface mesh using standard volume meshing techniques (frontal and Delaunay meshers are available in Gmsh).

The automatic procedure is implemented within the open source mesh generator Gmsh [2]. Examples of how to use it can be found on the Gmsh's wiki: `https://geuz.org/trac/gmsh`¶.

¶Access the wiki with username *gmsh* and password *gmsh*

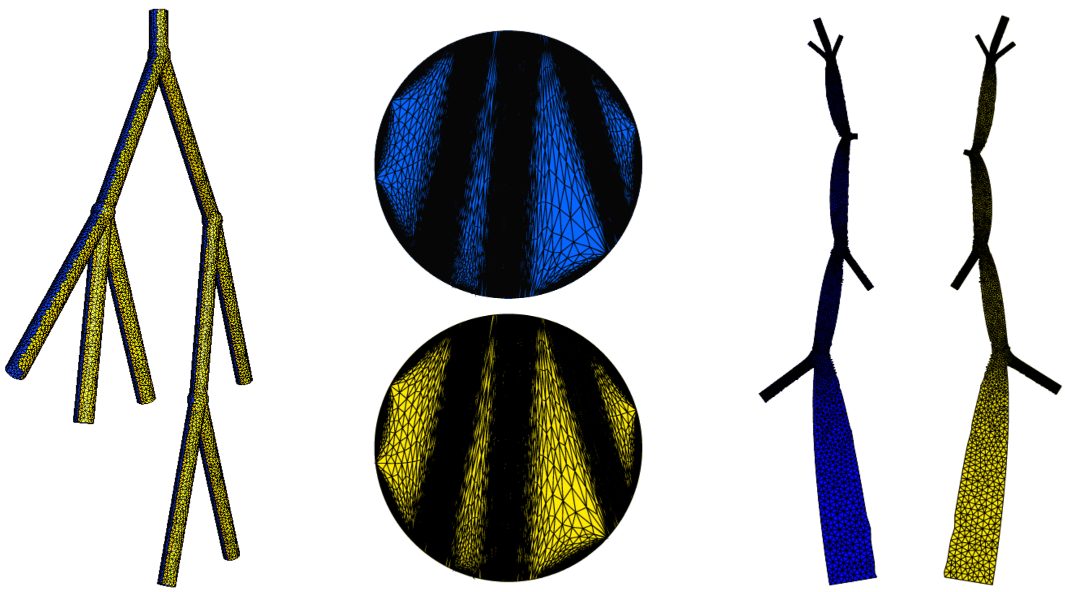


Figure 8. Remeshing of an arterial tree ($G = 0, N_B = 7, \eta = 28$). The initial mesh is first split into two parts using the multiscale Laplacian partitioning method. Each of those two parts is then mapped in the parametric space by computing (a) a Laplacian harmonic map (b) and a conformal map (c).

5. Results

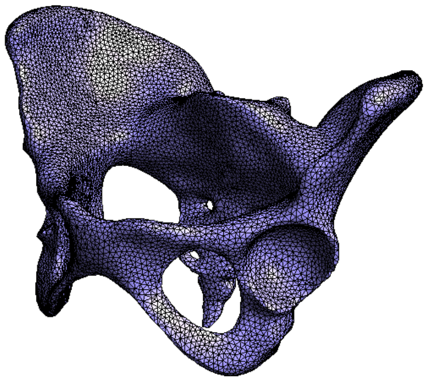
The first example on Fig.8 shows two different types of harmonic mappings (Laplacian and conformal) for an idealized arterial tree. The geometry has genus $G = 0$ and $N_B = 9$ boundaries, so hat the topological conditions are satisfied. However, the geometrical aspect ratio η is too high $\eta = 28$. Therefore, the input mesh has been partitioned into two parts with our multiscale Laplacian partitioning method. The mapped meshes are presented in Figs. 8b) and c). The mapped mesh obtained with the Laplacian harmonic map onto a unit disk presents lots of mesh distortion while the one obtained with the conformal mapping gives a less distorted parametrization and mesh metric. The remeshing being performed in the parametric space, it is important to keep in mind that the 2D meshing algorithms are more efficient with less distorted mesh metrics. Meanwhile our Gmsh's 2D meshing algorithms (Gmsh MeshAdapt, Delaunay and Frontal) are able to deal with a mesh distortion such as presented in Fig.8b.

The next example presented in Fig. 9 illustrates a uniform remeshing of a human pelvis that has genus $G = 9$ and that is a closed surface , i.e. $N_B = 0$. We compare the quality of the initial STL triangulation (obtained through a segmentation procedure) with the quality of the remeshed pelvis based on a laplacian harmonic map. The quality of the isotropic meshes is evaluated by computing the aspect ratio of every mesh triangle as follows [2]:

$$\kappa = \alpha \frac{\text{inscribed radius}}{\text{circumscribed radius}} = 4 \frac{\sin \hat{a} \sin \hat{b} \sin \hat{c}}{\sin \hat{a} + \sin \hat{b} + \sin \hat{c}}, \tag{6}$$

Prepared using nmeauth.cls

$\hat{a}, \hat{b}, \hat{c}$ being the three inner angles of the triangle. With this definition, the equilateral triangle has $\kappa = 1$ and degenerated (zero surface) triangles have $\kappa = 0$. For the example of the pelvis, it is observed that the mean $\bar{\kappa}$ and minimum quality κ_{min} of the new mesh are both very high: $\bar{\kappa} = 0.94, \kappa_{min} = 0.62$. Besides, the mean quality measure is found to be constant ($\pm 2\%$) for all examples and hence independent of the initial triangulation and the mesh density.



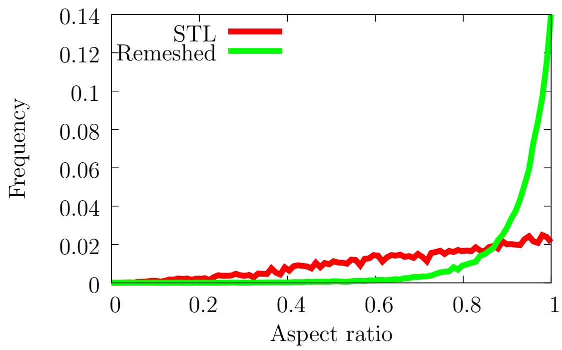


Figure 9. Remeshing of a human pelvis with a harmonic map ($G = 8, N_B = 0$). The left figure shows the remeshed surface and the right figure the quality histogram of both the initial STL file and the remeshed surface based on a Laplacian harmonic map.

The initial STL triangulation is made of 25.836 triangles and the remeshed human pelvis is composed of about the same number of elements (26.313 triangles). Figure 10 shows a pie chart diagram of the time spent in the different steps of the automatic remeshing procedure described in section 4. As can be seen in the diagram, most of the time is spent in the 2D meshing algorithms. The time needed for the parametrization computed with harmonic maps is only 5% of the total time, while the percentage of time needed for the partitioning (combination of the multiscale and multilevel partitioning methods) is less than 20% for this mesh. The total time for remeshing is $9s$ (on a MACBOOK PRO clocked at 2.4 GHz.) for this example. We also

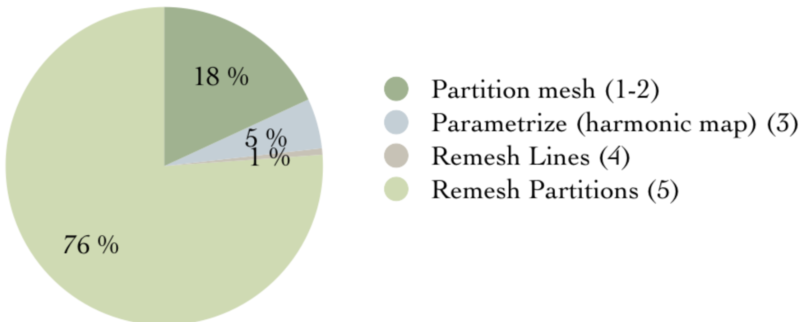


Figure 10. Time spent in the different steps of the automatic remeshing procedure described in section 4 fr th mesh model of the human pelvis $25k$ triangles. The total time for remeshing is $9s$.

Remeshing	Number of partitions	Partition time (s)	Parametrization time (s)	Total remeshing time (s)
LSCM Levy [7] (1.3Ghz)	23	30	95	–
Eck [3] (1.3Ghz)	88	-	-	33.5
ABF++ Zayer [22]	2	-	13	-
LinABF Zayer [22]	2	-	2	-
Presented method (2.4Ghz)				
* laplacian partitioner	2	16.7	1.4	25
* multilevel partitioner	10	7	1.4	14

Table I. Remeshing statistics and timings for the bunny mesh model of $70k$ triangles (new mesh of $25k$ triangles). Comparison (when available) of the presented method with other techniques presented in the Computer Graphics community.

compare the proposed method with two other remeshing packages presented in the literature. We first consider the well-known bunny mesh model of $70k$ triangles presented in Fig.11[‖]. The original mesh has $N_B = 5$ holes and its genus is $G = 0$. We compare in table I some statistics and timings of our algorithm with the least square conformal map (LSCM) of Levy et al. [7], with the multiresolution remeshing of Eck et al. [3] and with the angle based parametrization (ABF) of Zayer [22]. The table as well the timings for our remeshing procedure considering the two different partitioners.

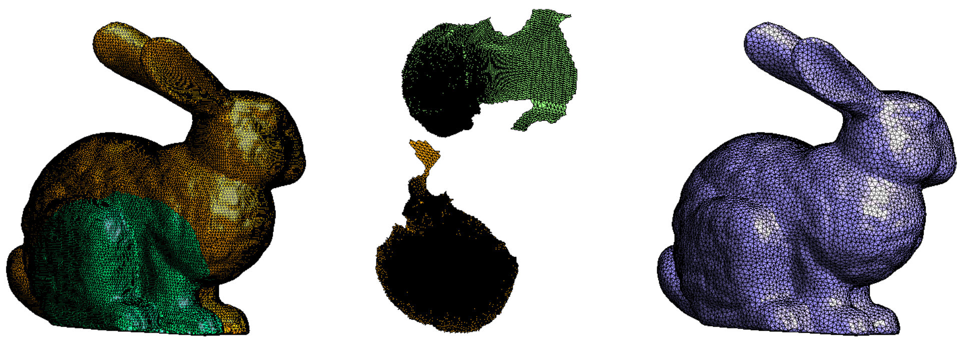


Figure 11. Remeshing of the bunny mesh model of $70k$ triangles ($G = 0, N_B = 5$). Left figure shows the two partitions, middle figure shows the conformal harmonic parametrization that has been computed for both mesh partitions and right figure shows the remeshed bunny with about $25k$ triangles.

Another standard test case that is used in the litterature is the feline mesh model of $50k$ vertices ($G = 0, N_B = 0$) presented in Fig. 12. Statistics on the remeshing procedure are presented in table II. The first remeshing package we compare with is a remeshing method

[‖]The model can be downloaded at the following web site:
http://www.sonycsl.co.jp/person/nielsen/visualcomputing/programs/bunny-conformal.obj

Prepared using nmeauth.cls

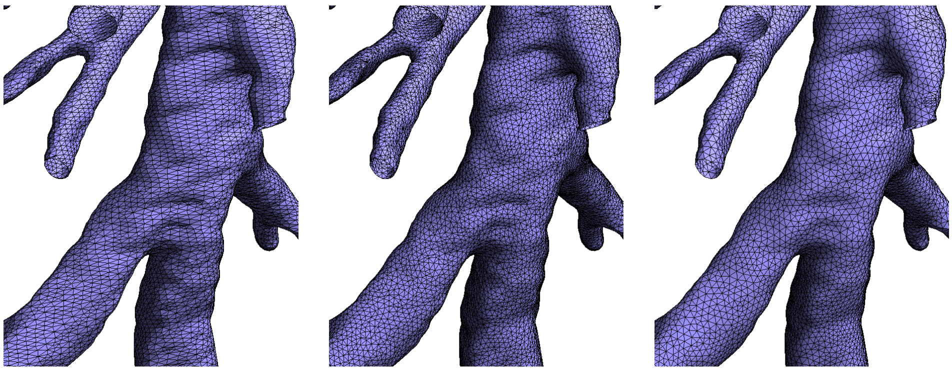
Mesh	Vertices	Min θ	Mean θ	L_2 Error	Remeshing time(s)
Original in [23, 24]	49 864	3.8°	40°	-	
Remeshed in [23, 24]	10 825	7.4°	48.3°	0.64%	74
Remeshed in [25]	256	–	–	–	900
Remeshed with presented method	12 946	11.6°	48.7°	0.53%	19

Table II. Remeshing statistics for the feline mesh model of $50k$ vertices ($95k$ triangles) with the remeshing time (new mesh of $25k$ triangles).

based on a local parametrization approach [23, 24] and the second approach is based on a global parametrization approach with an atlas of open base charts that covers the base mesh [25]. Table II shows the statistics of the re-meshes of the feline model along with the computational time for the remeshing procedure. The statistics are given in terms of the minimal angle of the triangles. For a high-quality mesh, the minimum of these values should be no less than $\theta = 10°$, and the average should be no less than $\theta = 45°$. We also present the geometrical remeshing error as the Hausdorff distance normalized by the bounding box diagonal, obtained using the Metro tool [26] and show the remeshing time. Results in table II show that our approach, besides from generating meshes with the highest quality with respect to all criteria, is more efficient in terms of computational time than the two other approaches.

Figure 12. Remeshing of the feline model ($G = 0, N_B = 0$) of $50k$ vertices with the presented method. Left figure shows the initial mesh that has been automatically partitionned into 13 patches and right figure shows the remeshed feline model.

Optimization of surface triangulations is very important in the context of biomedical geometries where the triangulations are constructed mostly from medical images through a segmentation procedure. We further illustrate the capabilities of our algorithm by showing the remeshing of airways models presented in Figs.13 and 14. The presented algorithm is compared with the meshing algorithm of Mimics. Mimics uses a two step mesh adaptation strategy in order to optimize the initial STL triangulation: in a first step three iterations of remeshing improve skewness to a minimal value of 0.4. In this step the maximal edge length is set to $0.5\,mm$ and the maximal geometrical error to $0.01\,mm$. Hereafter a quality preserving triangle reduction is performed. Again three iterations are done with a maximal edge length set to $0.5\,mm$ and the maximal geometrical error set to $0.05\,mm$. This provides the final remeshed model with Mimics. In comparison, our technique relies on a multiscale partitioning of the airway models into two parts of moderate geometrical aspect ratio. Each of those two parts is then parametrized using a harmonic map and the remeshing is then performed in the parametric plane using standard 2D meshing algorithms (MeshAdapt, Delaunay or frontal). As can be seen in Figs.13 and 14, the quality of the meshes obtained with a remeshing procedure based on a harmonic map is much higher than with the mesh adaptation strategies (such as the one implemented in Mimics).



a) b) c)

Figure 13. Remeshing of human lungs: a) part of the initial STL triangulation, b) remeshed geometry with Mimics (after 2 steps) and c) remeshed lung based on the Harmonic mapping remeshing procedure.

The next example presented in Fig. 15 shows parts of the initial STL mesh of a complete aorta illustrated in Figs. 5 and 6 and the mesh after the remeshing procedure. The remeshing procedure aims at removing the very small elements present in the initial mesh at the junctions of the arteries. The remeshing has been performed by first partitioning the mesh using the multiscale partitioning method. The two resulting mesh partitions are then suitable for the computation of harmonic maps and the remeshing procedure is finally performed in the parametric space.

Remeshing with parametrizations based on harmonic maps can also be very interesting for computational mechanics. In many cases, the surfaces are designed using a CAD system composed of multiple patches. When their sizes are of the same order of magnitude as the mesh size, those patches can result in very distorted mesh elements and impact dramatically on the quality of the boundary layer mesh and, hence, the CFD solution. For those small patches, the remeshing based on cross-patch parametrization enables to remove the small elements present in the cad-based meshes.

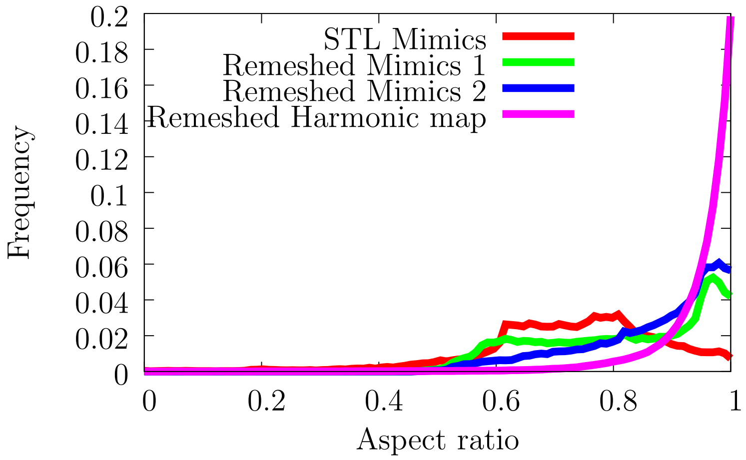


Figure 14. Remeshing of human lungs with the presented algorithm as compared with a commercial package such as Mimics.

Figure 16a shows a zoom of an initial mesh of a landing gear consisting of about 10^6 triangular mesh elements composed of 852 different patches. For this example, the small elements presented in the initial cad-based surface mesh (see for example Fig.17a) prevented us to build any CFD volume boundary layer mesh. Thanks to the cross-patch parametrization (Figs. 16b and 17b), we were able to reduce the number of patches to only 291 surface patches and build a suitable CFD boundary layer mesh of 12M nodes for that model (Fig. 19).

Figure 18 shows the the quality histograms of the two surface meshes: the cad-based mesh and remeshed mesh. We can see that the very small elements have been removed during the remeshing procedure. A preliminary computation has been done using the in-house flow solver developped at Cenaero [27]. The flow Reynolds number based on the diameter of the shock strut is $Re = 73000$ and the Mach number is $Ma = 0.166$. The flow is supposed to be fully turbulent. As the mesh is not fine enough to capture correctly all the turbulent scales, a Delayed Detached-Eddy Simulation [28] (DDSS) turbulence model is used. The DDES is a hybrid RANS (Reynolds-Averaged Navier Stokes)/LES (Large Eddy Simulation) method which uses a RANS modelling in the attached boundary layer and a LES model for the detached flow. The simulation ran on 200 CPU and the time needed to compute one convective time ($t_c = L/U_c$) is 3 hours. Fig. 20 shows the instantaneous pressure on the surface of the landing gear. An improvement of the volume mesh is in progress in order to obtain a better representation of the wake. The results will be used to measure, with an acoustic solver, the sound created by the landing gear. This case has been studied within the frame of the BANC (Benchmark problems for Airframe Noise Computations) Workshop.

The definition of the patches to be cross-parametrized is performed by the user but takes few time regarding the complexity of the CFD computations. For the industrial example of the landing gear, the definition of the new patches and the remeshing takes only 1 hour on 1 CPU compared to 3 hours on 200 CPU for the CFD simulation. Besides, as the remeshing method based on cross-patch parametrization is fast and simple, it can be easily integrated and automated in an industrial CFD computation chain.

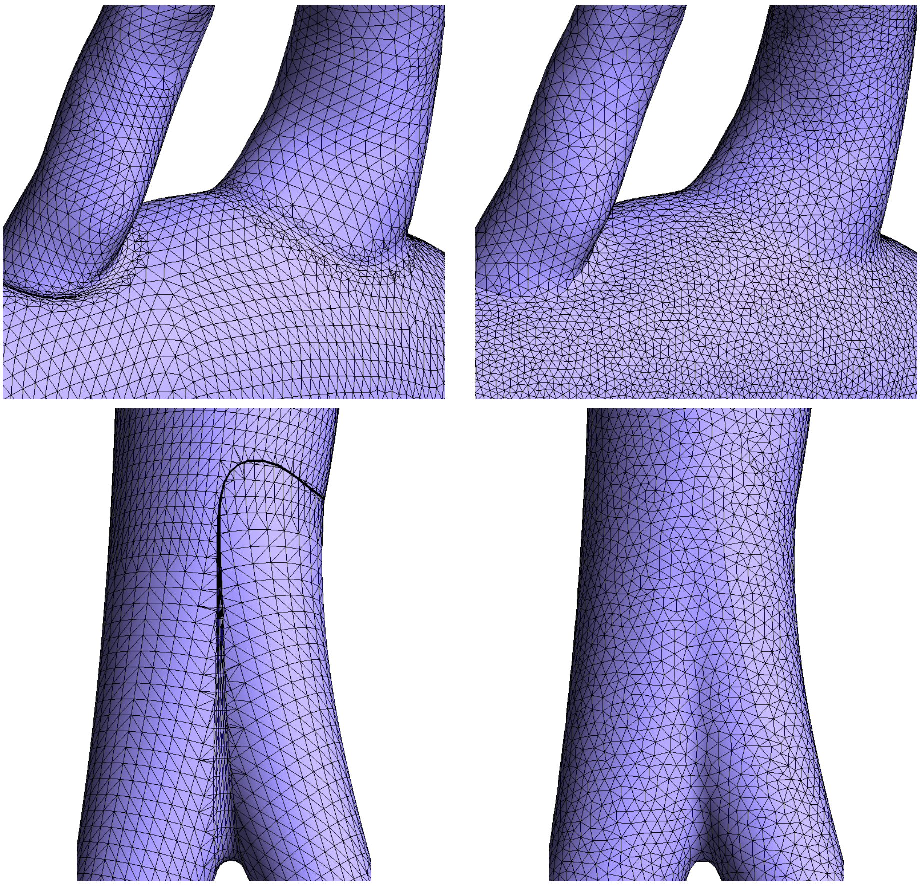
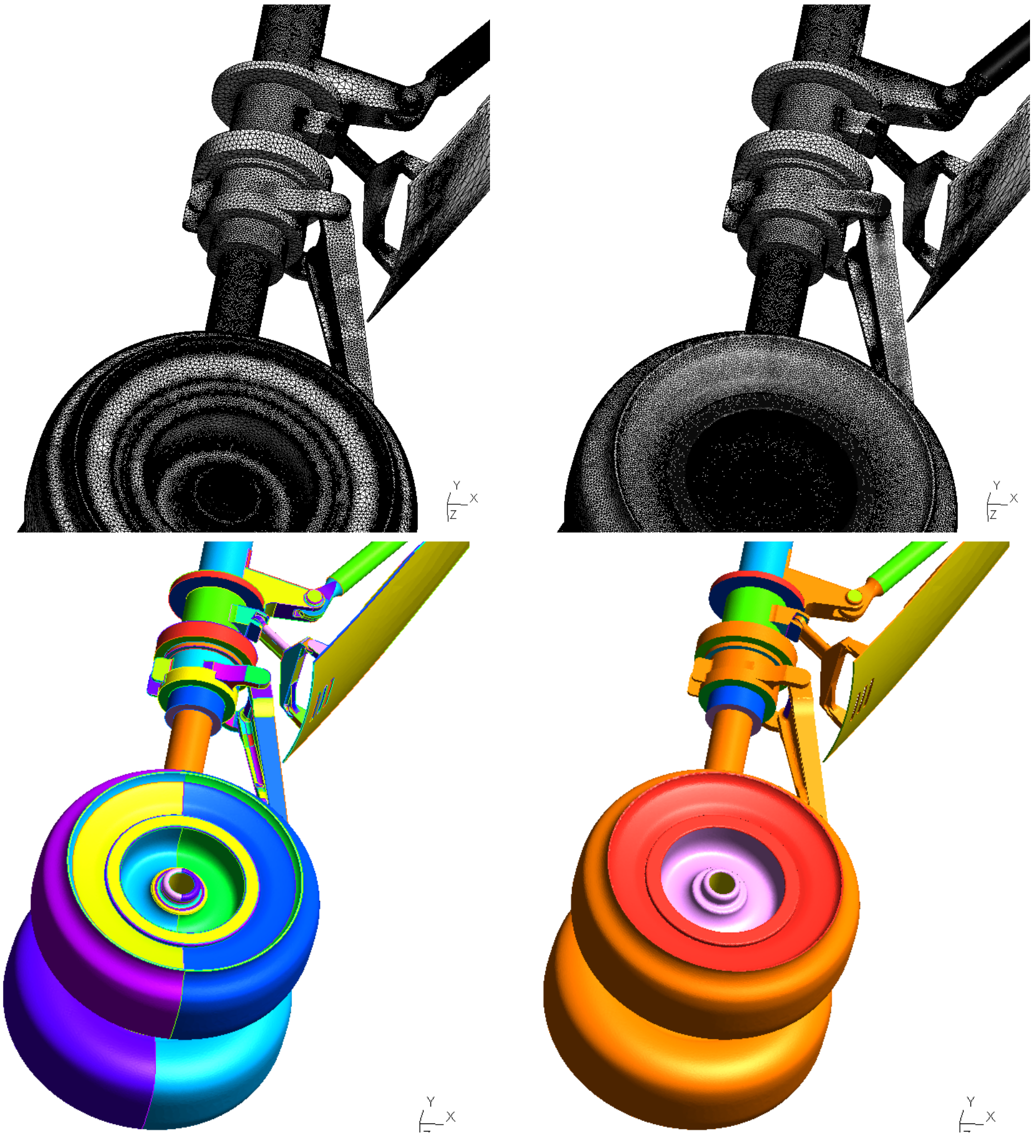


Figure 15. Remeshing of a the aortic artery presented in Fig. 5. We show parts of the mesh near the bifurcations before (left figures) and after (right figures) the remeshing procedure based on a Laplacian harmonic map.

6. Conclusion

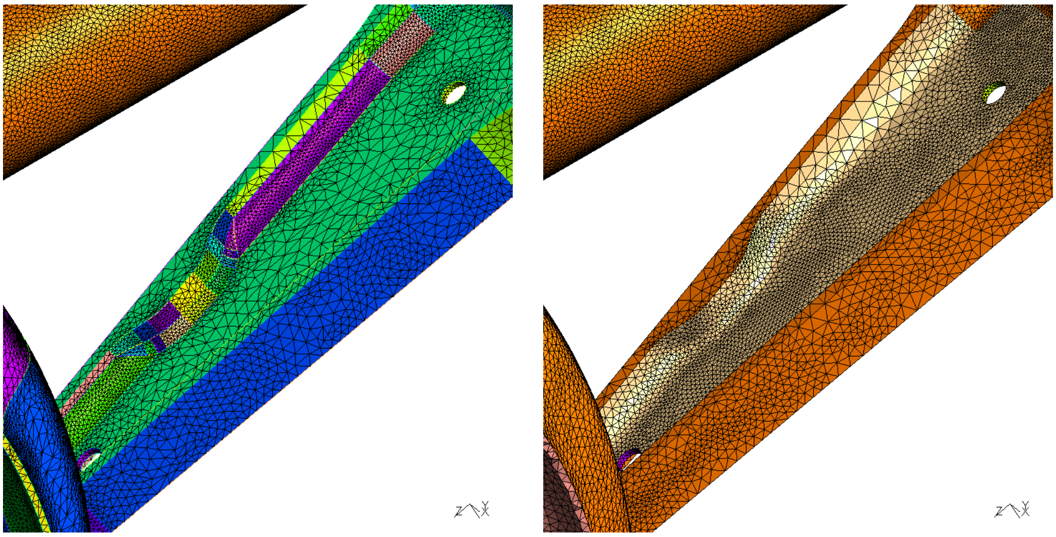
In this work, we have presented a fully automatic approach based on harmonic mappings for remeshing surfaces that overcomes the limitations of harmonic maps: namely limitations on the geometrical aspect ratio and on the genus of the surface. The approach is original as it combines an novel multiscale harmonic partitioning method with a multilevel partitioning algorithm to come up with an automatic remeshing algorithm that overcomes the limitations of harmonic maps. With the present approach, we are able to remesh any surface with any topological genus and with large geometrical aspect ratios. We showed that the remeshing procedure produces high-quality meshes and that it can be used for fairly complex biomedical and industrial applications.



a) b)

Figure 16. Cross-patch remeshing of a landing gear (with patches of $G = 0$, $N_B = 1$). The left figures a) show a surface mesh that is made of 582 geometrical patches and the right figures b) show the remeshed surface made of only 291 patches. The remeshed patches are computed using cross-patch parametrization.

ACKNOWLEDGEMENTS



a) b)

Figure 17. Remeshing of a landing gear. Left Figure shows a part of the initial mesh with many patches and right Figure shows the mesh with the reparametrized patches.

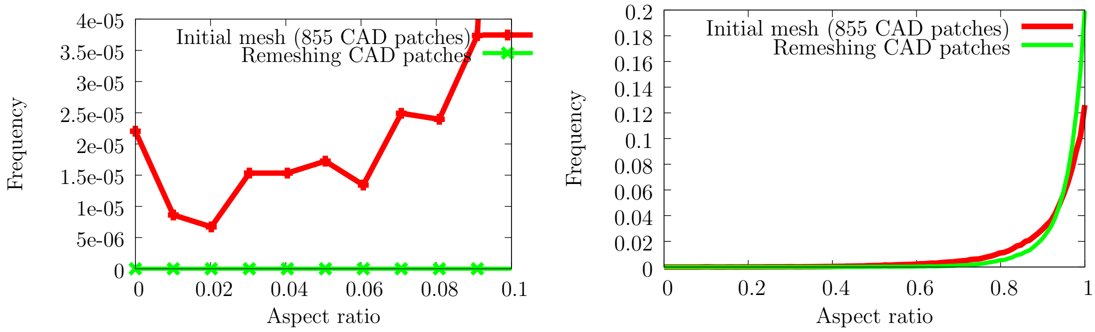


Figure 18. Quality histogram for the remeshing of a landing gear. The left figure shows a zoom for range of small aspect ratio $\kappa \in [0:0.1]$ and the right figure the whole range of aspect ratio $\kappa \in [0:1]$. As can be seen, the remeshing procedure has removed the small elements by merging different patches using a cross-patch parametrization.

The authors gratefully acknowledge the Walloon Region (EFCONIVO Project) for their financial support. The authors would also like to thank the people from FluidDA n.v. for providing the airway models and their feedback in the analysis of the results.

REFERENCES

1. Remacle JF, Geuzaine C, Compère G, Marchandise E. High quality surface remeshing using harmonic maps. *International Journal for Numerical Methods in Engineering* 2010; **83**(403-425).

Figure 19. Remeshing of a landing gear. Figure shows part of the 3D mesh boundary layer mesh around the landing gear.

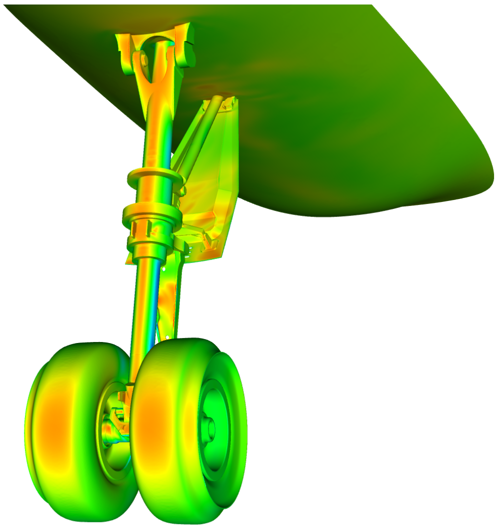


Figure 20. Instantaneous pressure field on the surface of the landing gear.

2. Geuzaine C, Remacle JF. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering* 2009; **79**(11):1309–1331.
3. Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W. Multiresolution analysis of arbitrary meshes. *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive*

techniques, 1995; 173–182.
4. Floater MS. Parametrization and smooth approximation of surface triangulations. *Computer aided geometric design* 1997; **14**(231-250).
5. Sheffer A, de Sturler E. Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers* 2001; **17**(3):1435–5663.
6. Sheffer A, Lévy B, Lorraine I, Mogilnitsky M, Bogomyakov E. Abf++: fast and robust angle based flattening. *ACM Transactions on Graphics* 2005; **24**(311-330).
7. Levy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. *Computer Graphics (Proceedings of SIGGRAPH 02)*, 2002; 362 – 371.
8. Alliez P, Meyer M, Desbrun M. Interactive geometry remeshing. *Computer graphics (Proceedings of the SIGGRAPH 02)* 2002; :347–354.
9. Mullen P, Tong Y, Alliez P, Desbrun M. Spectral conformal parameterization. *In ACM/EG Symposium of Geometry Processing*, 2008.
10. Alliez P, Meyer M, Desbrun M. Interactive geometry remeshing. *Computer graphics (Proceedings of the SIGGRAPH 02)* 2002; :347–354.
11. Shinagawa Y, Kunii T, Kergosien YL. Surface coding based on morse theory. *IEEE Computer Graphics and Applications* 1991; **11**(5):66–78.
12. Gu X, Gortler S, Hoppe H. Geometry images. *ACM SIGGRAPH*, 2002; 656–361.
13. Gu X, Yau S. Global conformal surface parameterization. *Proceedings of the first symposium on geometry processing*, 2003; 127–137.
14. Khodakovsky A, Litke N, Shröder P. Globally smooth parameterizations with low distortion. *ACM Transactions on Graphics* 2003; **22**(3):350–357.
15. Chang CT, Gorissen B, Melchior S. Fast computation of the minimal oriented bounding box on the rotation group so(3). *ACM Transactions on Graphics* 2009, to appear; .
16. Durand N, Alliot JM. A combined nelder-mead simplex and genetic algorithm. *GECCO'99: Proc. Genetic and Evol. Comp. Conf*, 1999; 1–7.
17. Floater MS. Parametric tilings and scattered data approximation. *International Journal of Shape Modeling* 1998; **4**:165–182.
18. Floater MS. One-to-one piecewise linear mappings over trinagulations. *Math. Comp* 2003; **72**(685-696).
19. Kaczynski T, Mischaikow K, Mrozek M. *Computational Homology*, *Applied Mathematical Sciences*, vol. 157. Springer, 2004.
20. Hendrickson B, Leland R. A multilevel algorithm for partitioning graphs. *In Proc. Supercomputing*, 1995.
21. Karypis G, Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs. *International Conference on Parallel Processing*, 1995; 113–122.
22. Zayer R, Lévy B, Seidel HP. Linear angle based parameterization. *ACM/EG Symposium on Geometry Processing conference proceedings*, 2007.
23. Surazhsky V, Alliez P, , Gotsman C. Isotropic remeshing of surfaces: a local parameterization approach. *in proceedings of 12th International Meshing Roundtable,*, Santa Fe, New Mexico, USA, 2003; 215–224.
24. Surazhsky V, Gotsman C. Explicit surface remeshing. *Eurographics Symposium on Geometry Processing*, vol. 17-28, L Kobbelt HHE P Schröder (ed.), 2003.
25. Guskov I. Manifold-based approach to semi-regular remeshing. *Graphical Models* 2007; **69**(1):1–18.
26. P Cignoni CR, 1998 RS. Metro : Measuring error on simplified surfaces. *Computer Graphics Forum* 1998; **17**(2):167–174.
27. Georges L, Winckelmans GS, Geuzaine P. Improving shock-free compressible rans solvers for les on unstructured meshes. *Journal of Computational and Applied Mathematics* 2008; **215**(2):419–428.
28. Spalart P, Deck S, Schur M, Squires K, Strelets M, Travin A. A new version of detached-eddy simulation, resistant to ambiguous grid densities. *Theoretical and Computational Fluid Dynamics* 2006; **11**(8):181–195.

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN BIOMEDICAL ENGINEERING
Int. J. Numer. Meth. Biomed. Engng. 2010; **83**:876–889
Published online 12 April 2010 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cnm.1388

Quality meshing based on STL triangulations for biomedical simulations

E. Marchandise[1,*,†], G. Compère[1], M. Willemet[1], G. Bricteux[1], C. Geuzaine[2] and J.-F. Remacle[1]

[1]*Institute of Mechanics, Materials and Civil Engineering (iMMC), Université catholique de Louvain, Avenue G.Lemaitre, 4 1348 Louvain-la-Neuve, Belgium*
[2]*Department of Electrical Engineering and Computer Science, Université de Liège, Montefiore Institute B28, 4000 Liège, Belgium*

SUMMARY

This work describes an automatic approach to recover a high-quality surface mesh from low-quality or oversampled inputs (STL-files) obtained from medical imaging through classical segmentation techniques. The approach combines a robust method of parametrization based on harmonic maps (*Int. J. Numer. Meth. Engng.* 2009; accepted) with a recursive call to a multi-level edge partitioning software. By doing so, we are able to get rid of the topological and the geometrical limitations of harmonic maps. The overall remeshing procedure is implemented, together with the finite element discretization procedure required for computing harmonic maps, in the open-source mesh generator Gmsh (*Int. J. Numer. Meth. Engng.* 2009; **79**(11):1309–1331). We show that the proposed method produces high-quality meshes and we highlight the benefits of using those high-quality meshes for biomedical simulations. Copyright © 2010 John Wiley & Sons, Ltd.

Received 30 November 2009; Revised 16 February 2010; Accepted 17 February 2010

KEY WORDS: surface mapping; surface meshing; parametrization; open source; remeshing; STL files; biomedical

1. INTRODUCTION

In the biomedical field, geometrical data are acquired through medical imaging techniques such as CT scan or MRI. The data are then usually given to end-users as an STL triangulation that comes as the output of a surface reconstruction algorithm applied to the point cloud obtained from the medical images [1]. Those generated STL triangulations can serve as input for most volume meshing algorithms [2, 3]. Yet, those STL triangulations are generally oversampled and of very low quality, with poorly shaped and distorted triangles. This is still to date a major bottleneck in the domain of biomedical computations since the quality of the mesh impacts both on the efficiency and the accuracy of numerical solutions [4, 5]. For example, it is well known that for finite element computations, the discretization error in the finite element solution increases when the element angles become too large [6], and the condition number of the element matrix increases with small angles [7]. It is then desirable to modify the initial surface mesh to generate a new

*Correspondence to: E. Marchandise, Institute of Mechanics, Materials and Civil Engineering (iMMC), Université catholique de Louvain, Avenue G.Lemaitre, 4 1348 Louvain-la-Neuve, Belgium.
†E-mail: emilie.marchandise@uclouvain.be

Contract/grant sponsor: Belgian National Fund for Scientific Research (FNRS)
Contract/grant sponsor: Walloon Region (EFCONIVO Project)

surface mesh with nearly equilateral triangles or with a smooth gradation of triangle density based on the geometry curvature. This procedure is called *remeshing*.

There exist mainly two approaches for surface remeshing: mesh adaptation strategies [8–10] and parametrization techniques [11–16]. The mesh adaptation strategies belong to the direct meshing methods and use local mesh modifications in order to both improve the quality of the input surface mesh and adapt the mesh to a given mesh size criterion. The parametrization techniques belong to the indirect meshing approach. The initial 3D surface mesh is first parameterized onto a 2D planar surface mesh; the initial surface can then be remeshed using any 2D mesh generation procedure by subsequently mapping the new mesh back to the original surface.

When a parametrization of the surface is available, it is usually better to use it for remeshing. Indeed, when a parametrization is available, ensuring that the surface mesh has the right topology is trivial. Also, as the medical geometries are often highly oversampled and are of very poor quality, the numerous sampling operations are much more efficient in the parameter plane than in 3D space.

In a recent paper [17], we have introduced an efficient approach for high-quality remeshing of surfaces based on a parametrization technique. The approach uses a discrete finite element harmonic map to parameterize the input triangulation onto a unit disk. By combining it with a local cavity check algorithm that enforces the discrete harmonic map to be one-to-one, we came out with a robust method for remeshing that is advantageous compared with mesh adaptation methods. However, as it was highlighted in [17], there are two important limitations of harmonic maps, namely limitations on the genus and the geometrical aspect of the surface. Indeed, to be able to parameterize the triangulation onto a unit disk, the triangulation should be homeomorphic to a disk, i.e should have a genus zero with at least one boundary. Besides, as the solution of harmonic maps tends exponentially to a constant, the triangulation should have a uniform geometrical aspect ratio to prevent non-distinguishable coordinates.

In this paper, we present a robust and automatic way to overcome the topological and the geometrical limitations of harmonic maps. The presented algorithm combines a discrete harmonic mapping with a multi-level edge partitioning software that recursively partitions the triangulation into a small number of charts that satisfy the topological and geometrical constraints. We show that our method renders high-quality meshes and highlight the benefits of using those high-quality meshes for cardiovascular and bone biomechanical simulations.

2. MESHING WITH HARMONIC MAPS

The key feature of our remeshing algorithm presented in [17] is to define a map that transforms continuously a surface $\mathscr{S} \in \mathscr{R}^3$ into a unit disk $\mathscr{S}'$ embedded in $\mathscr{R}^2$ [18, 19]. The parametrization should be a bijective function $\mathbf{u}(\mathbf{x})$:

$$\mathbf{x} \in \mathscr{S} \subset \mathscr{R}^3 \mapsto \mathbf{u}(\mathbf{x}) \in \mathscr{S}' \subset \mathscr{R}^2. \tag{1}$$

Such a parametrization exists if the two surfaces $\mathscr{S}$ and $\mathscr{S}'$ have the same topology, i.e are both zero genus surfaces ($G=0$) and have at least one boundary ($N_B \geqslant 1$).[‡] When the surface $\mathscr{S}$ is a triangular mesh as in the case of an STL file, the genus can be easily computed from the Euler–Poincare formula:

$$G = \frac{-N_\mathrm{V} + N_\mathrm{E} - N_\mathrm{T} + 2 - N_B}{2}, \tag{2}$$

where N_V, N_E and N_T are, respectively, the number of vertices, edges and triangles.

Harmonic maps have been chosen for the parametrization [20, 21], by solving one Laplace problem for each coordinate:

$$\nabla^2 u = 0, \quad \nabla^2 v = 0, \tag{3}$$

[‡]For example, a sphere has $G=0$ and $N_B=0$ and a torus has $G=1$ and $N_B=0$.

DOI: 10.1002/cnm

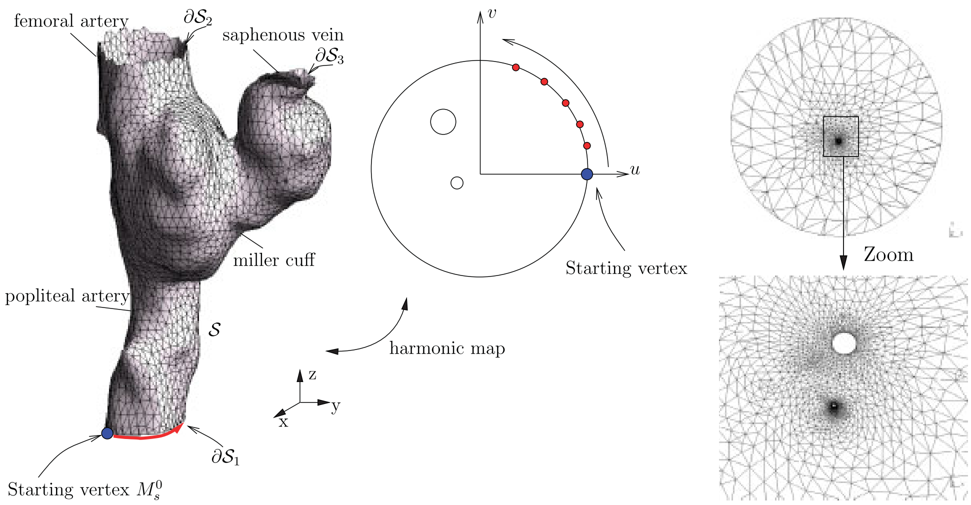


Figure 1. STL triangulation of an arterial anastomosis ($G=0, N_B=3, \eta=5$) and its map onto the unit circle (Left) and mapped mesh on the unit circle (Right). As the geometrical ratio of the initial STL triangulation is higher than 4, the mapped triangles become very small (see zoom) in the parametric unit disk.

with appropriate Dirichlet boundary condition for one of the boundaries $\partial\mathscr{S}_1$ of the surface $\mathscr{S}$,

$$u(l)=\cos(2\pi l/L), \quad v(l)=\sin(2\pi l/L), \tag{4}$$

and with Neumann boundary conditions for the other boundaries. In (4), l denotes the curvilinear abscissa of a point along the boundary $\partial\mathscr{S}_1$ of total length L.

The discrete harmonic map is obtained through a finite element formulation of the Laplace problem (3)–(4) on the STL triangulation. The finite element solutions provide to each internal vertex of the original triangulation $\mathbf{x}$ its local coordinates u and v. However, as shown in [17] the solution of the mapping becomes exponentially small[§] for vertices located away from $\partial\mathscr{S}_1$. As a consequence, local coordinates u and v of those far away vertices might numerically become indistinguishable (see the zoom in Figure 1). To prevent this, the geometrical aspect ratio of the surface

$$\eta=\frac{H}{D} \tag{5}$$

should be smaller than 4. Indeed, we can show that $\eta=4$ corresponds to an area of mapped triangles of about $r_i^2=10^{-10}$ (see Equation 23 and Figure 10(c) in [17]). In (5), H is the maximal distance (computed on the 3D surface $\mathscr{S}$) of a mesh vertex to the boundary $\partial\mathscr{S}_1$ and D is the equivalent diameter of the boundary $\partial\mathscr{S}_1$.

Figure 1 shows both an initial triangular mesh of $\mathscr{S}$ and its map onto the unit disk. The surface $\mathscr{S}$ results from the segmentation of an anastomosis site in the lower limbs, more precisely a bypass of an occluded femoral artery realized with the patient's saphenous vein. The unit disk D contains two holes that correspond to the boundary of the femoral artery $\partial\mathscr{S}_2$ and the saphenous vein $\partial\mathscr{S}_3$ on which we have imposed Neumann boundary conditions.

Once the parametrization is computed, we use standard 2D anisotropic mesh generation procedures onto the unit disk, with the aim of producing a mesh in the real 3D space that has controlled element sizes and shapes. In order to control the surface element sizes, we define an isotropic mesh size field [22] $\delta(\mathbf{x})$ that is a function that gives the optimal mesh edge length at point $\mathbf{x}$.

[§]In principle, the solution becomes constant far from the boundary, this constant being the average of the solution on the boundary. Yet, the average of the solution on the boundary being zero, the solution goes to zero far from the boundary.

DOI: 10.1002/cnm

In the examples that will be presented, the mesh size field is chosen to be either a constant or varies according to the curvature of the geometry.

3. AUTOMATIC QUALITY REMESHING

In the previous section, we put to the fore the topological and geometrical limitations of harmonic maps. To sum up, for the proposed discrete harmonic maps we need

(i) $G=0$;
(ii) $N_B \geqslant 1$;
(iii) $\eta<4$.

The first condition can be verified using Equation (2); the second condition can be checked by looking simply at the topology of the mesh. The third condition is less trivial to assess.

In the computer graphics community, people overcome all three conditions simultaneously by using a partition scheme based on the concept of Voronoi diagrams [21] or inspired by the Morse theory [16, 23]. The resulting mesh partitions are area-balanced patches that satisfy the three conditions. However, this approach results in a large number of patches and hence a large number of interfaces between those patches, which are not desirable.

We propose in this paper a fast and automatic way to overcome both topological and geometrical limitations of harmonic maps. The idea is to combine an harmonic map with a multilevel edge partitioning softwares such as Chaco [24] or Metis [25] to partition recursively the triangulation into a minimal number of partitions that satisfy the topological and geometrical conditions. Multilevel methods are attractive since they reduce the costs of spectral partitioning methods while still generating high-quality partitions. These work on the connectivity graph of the mesh, but instead of trying to split this directly, the graph is first condensed through a number of levels. The condensation is achieved through clustering together vertices that are closed together to produce a graph with fewer vertices. New edges between the clusters are weighted to reflect the number of edges that existed in the larger graph. By using several levels of condensation a much smaller graph can be obtained that is easily partitioned by a method such as spectral bisection. This partitioning information can then be transferred up through the levels to the original graph.

The automatic procedure for a uniform remeshing a triangulation $\mathscr{S}$ with prescribed mesh size δ is illustrated in Figures 2 and 3 and reads as follows:

1. Check conditions (i)–(iii). If those conditions are not satisfied, recursively split the mesh with the multi-level partitioning software until satisfied. The geometrical aspect ratio η is computed approximately by using the ratio between the maximal size of the bounding box of the mesh partition and the maximal size of the bounding box of the boundaries $\partial\mathscr{S}$ of the mesh partition [26] (see illustration in Figure 3(1));
2. Remesh the lines that are the boundaries of the triangulation and the interfaces between the mesh partitions (see the interfaces between colored patches in 2(a)) that are represented by highlighted white lines in Figure 2(b)); Those lines are defined as model edges and divided into N parts as follows: $N=\int_0^L \|\mathbf{x}_{,t}\|/\delta\,\mathrm{d}t$. The remeshed lines are embedded in the final mesh shown in Figure 2(c)).
3. Compute the harmonic mapping for every mesh partition as explained in the previous section. If the boundary is composed of several parts $\partial\mathscr{S}_i$, assign the Dirichlet boundary conditions (4) to the closed boundary that has the largest bounding box.
4. Use standard surface meshers to remesh every partition in the parametric space and map the triangulation back to the original surface.
5. If a volume mesh is needed, generate a volume mesh from the new surface mesh using standard volume meshing techniques.

In our algorithm, the bounding boxes are oriented bounding boxes that are computed with the fast Oriented bounding box HYBBRID optimization algorithm presented in [26], which combines the genetic and the Nelder–Mead algorithms [27].

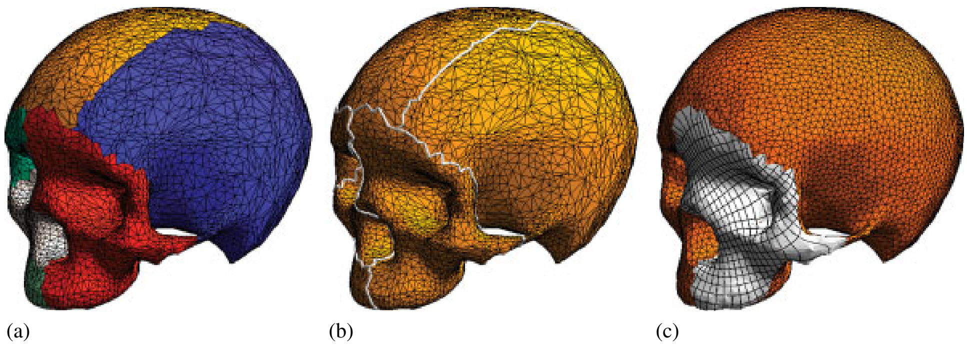


Figure 2. Remeshing algorithm: (a) initial triangulation ($G=2, N_B=0$) that is cut into different mesh partitions of zero genus; (b) remesh the lines at the interfaces between partition; and (c) compute harmonic map for every partition and remesh the partition in the parametric space ($\mathbf{u}(\mathbf{x})$ coordinates visible for one partition).

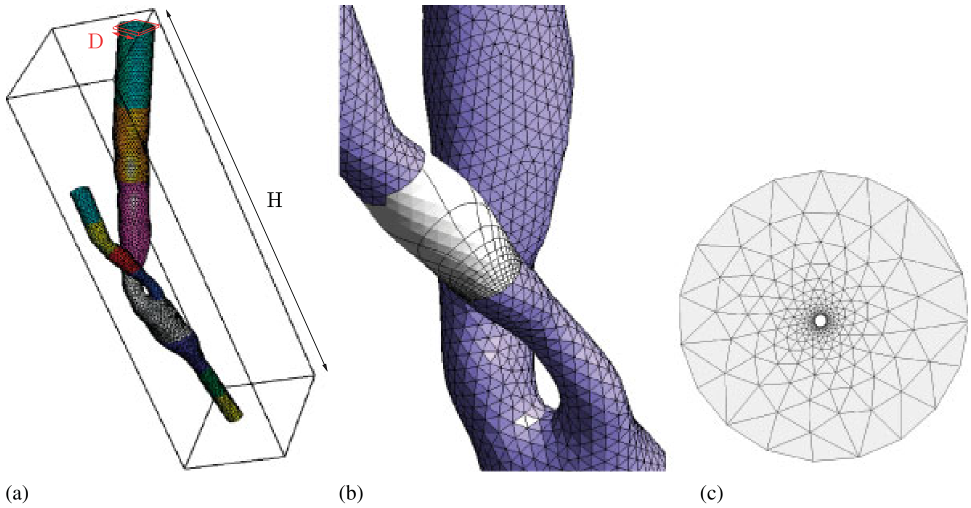


Figure 3. Remeshing algorithm: (a) initial triangulation ($G=0, N_B=3, \eta=H/D=16$) that is cut into different mesh partitions of uniform geometrical aspect ratio; (b) the harmonic map is computed for every partition ($\mathbf{u}(\mathbf{x})$ coordinates visible for one partition); and (c) remesh every partition in the parametric space. The mapped initial triangulation is shown for the partition visible on the middle image.

The automatic procedure is implemented within the open-source mesh generator Gmsh [22]. We show a simple example of how to use it. We suppose that we have an initial surface mesh and write the following geometry file 'remesh.geo':

```
// Merge the STL triangulation
Merge "skull.stl";

// Remesh the edges (if any), and faces with the presented algorithm
Compound Surface(100) = {1};

// Create a volume and mesh given the new surface mesh
Surface Loop(2) = {100};
Volume(3) = {2};
```

Other examples can be found on the Gmsh wiki: https://geuz.org/trac/gmsh (username: gmsh, password: gmsh).

4. RESULTS

4.1. High-quality surface and volume meshes

We have run our computational algorithm on a variety of medical geometries of arbitrary genus and complexity. Figure 4 illustrates a uniform remeshing of, respectively, a skull, an upper jaw and a hemipelvis. The top figure shows the remeshing of a human skull, the middle figures the remeshing of an upper jaw that is oversampled (116 k vertices) and the lower figures the remeshing of an initial poor-quality mesh of an hemipelvis. None of those initial triangulations satisfy the topological conditions: the skull has genus $G=2$, the jaw has genus $G=0$ but has $N_B=0$ and the pelvis has $G=1$ and $N_B=0$.

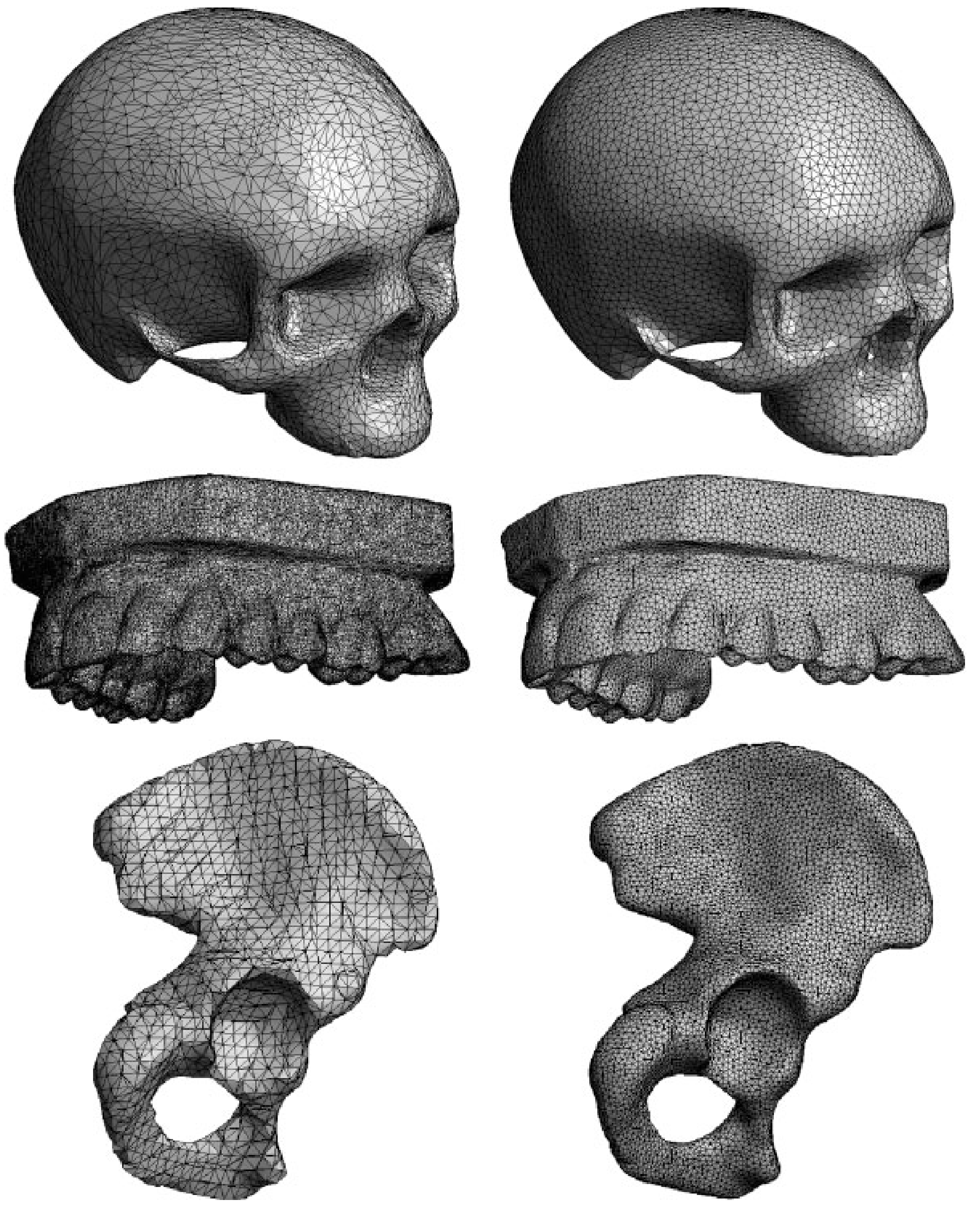


Figure 4. STL triangulations obtained from medical images (Left) that have been automatically remeshed with our automatic remeshing algorithm (Right).

DOI: 10.1002/cnm

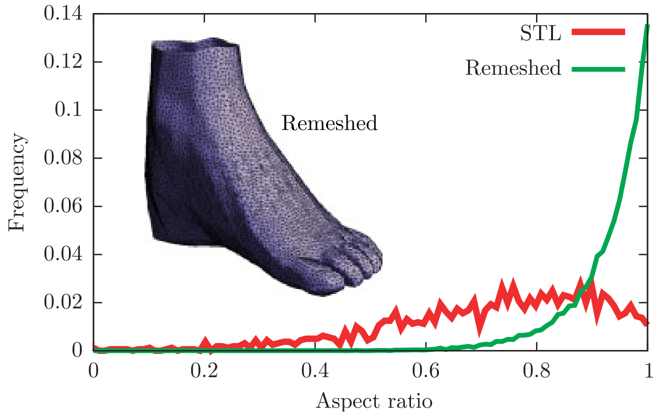


Figure 5. Plot of the quality histogram of both the STL triangulation and the remeshed surface of a scanned foot.

The quality of an isotropic mesh is evaluated by computing the aspect ratio of every mesh triangle as follows [22]:

$$\kappa = \alpha \frac{\text{inscribed radius}}{\text{circumscribed radius}} = 4 \frac{\sin\hat{a}\sin\hat{b}\sin\hat{c}}{\sin\hat{a}+\sin\hat{b}+\sin\hat{c}}, \tag{6}$$

$\hat{a}$, $\hat{b}$, $\hat{c}$ being the three inner angles of the triangle. With this definition, the equilateral triangle has $\kappa = 1$ and degenerated (zero surface) triangles have $\kappa = 0$.

Figure 5 shows the quality histogram for the initial triangulation of a foot and the remeshed geometry. As seen in Figure 5, the mean $\bar{\kappa}$ and minimum quality κ_{min} of the new mesh are both very high: $\bar{\kappa} = 0.94$, $\kappa_{\text{min}} = 0.62$. This mean quality measure was found to be constant ($\pm 2\%$) for all examples and hence independent of the initial triangulation and the mesh density. Volume tetrahedral meshes can then be created from those surface meshes. In order to measure the quality of the tetrahedral elements, we define another quality measure γ based also on the element radii ratio [22, 28]:

$$\gamma = \frac{6\sqrt{6}V}{S_{\text{F}} L_{\text{E}}},$$

V being the volume of the tetrahedron, S_{F} being the sum of the areas of the four faces of the tetrahedron and L_{E} being the sum of the lengths of the six edges of the tetrahedron. This γ quality measure lies in the interval [0,1], an element with $\gamma = 0$ being a sliver (zero volume). When creating volume meshes from surfaces that have been remeshed with our algorithm, we obtain also quite constant γ qualities, i.e $\gamma_{\text{min}} = 0.25 \pm 10\%$ and $\bar{\gamma} = 0.7 \pm 10\%$. This is much better than the gamma quality of volume meshes created from STL triangulations. Indeed the quality of those volume meshes is often very poor, with elements being small slivers $\gamma_{\text{min}} < 1 \cdot \text{e}^{-5}$ that will hinder or event prevent the convergence of the numerical method.

The time necessary to generate with our algorithm a new surface mesh less is less than 100 s for 10^6 elements.

4.2. *Quality meshing for biomedical simulations*

The two first biomedical simulations concern blood flow simulations. In the first example, blood flow in a distal anastomosis of a bypass is considered. While this problem has often been studied *in vitro* in simplified geometries [29–31], the simulation of blood flow in *in vivo* complex geometries is of great interest when one wants to focus on the patient-specific aspect [32]. As this is not the goal of this study, we refer the reader to the cited references for detailed hemodynamical analysis. We intend to illustrate in the following test case that in real and complex geometries, a

DOI: 10.1002/cnm

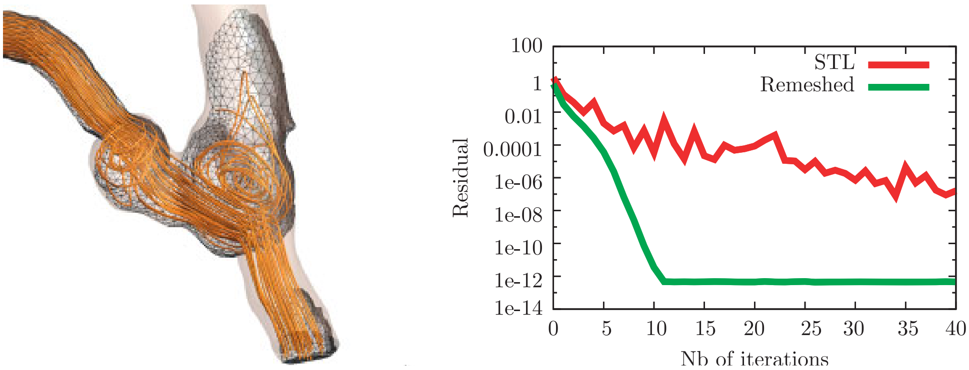


Figure 6. Blood flow simulation in an arterial bypass. The left figure shows the streamlines (zoom near the anastomosis) and the right figure shows the residual decrease for the two different volume meshes.

Table I. Quality of the surface and volume meshes.

	Surface quality		Volume quality	
Mesh	$\kappa_{\min}$	$\overline{\kappa}$	$\gamma_{\min}$	$\overline{\gamma}$
STL	0.0033	0.821	0.0019	0.563
Remeshed	0.6400	0.949	0.2550	0.677

high-quality mesh is required in order to ensure the numerical convergence of the simulation. Identical conclusions have been reached in computational studies related to other biomedical applications and considering the effects of various meshing styles: Vinchurkar and Longest [33, 34] have shown the performances of different types and qualities of meshes in the complex branching geometries of the respiratory system; Liu *et al.* [35] compared simulations in the total cavopulmonary connection using structured and unstructured meshes; Ethier and Prakash [36] studied a mesh convergence study of blood flow in a coronary artery model.

In the following two test cases, blood flow is governed by the incompressible Navier–Stokes equations for a Newtonian fluid. We use an implicit pressure-stabilized finite element method that has been shown to be robust, accurate and stable [37]. The linearized system is solved by using a GMRES solver with a relative convergence tolerance of 10^{-12}. The fluid properties of blood are taken to be $\rho = 1060\,\mathrm{kg/m^3}$ for the density and $\mu = 3.5\,\mathrm{Pa\,s}$ for the dynamic viscosity.

The first test case studies steady blood flow at Reynolds $Re = 900$ (based on the inlet diameter and average inlet velocity) in a veinous anastomosis of an occluded femoral artery (Figure 6) [38]. The anastomosis is segmented out of raw image data from a patient who underwent lower-limb bypass surgery. Two surface meshes are produced, one from the initial STL triangulation and the other using the remeshing procedure based on harmonic maps. Those surface meshes then serve as input for the generation of two volume meshes of about 10^4 tetrahedra (an STL-based and a remeshed-based volume mesh). Table I shows the quality of these two surface and volume meshes: the remeshed mesh presents higher minimal and mean qualities that enable the flow solver to converge better (see Figure 6).

The simulation is run with a constant flow rate at the inlet ($\overline{Q} = 75\,\mathrm{ml/min}$), a no-slip boundary condition at the walls and a constant pressure boundary condition on the outlet surface ($p = 50\,\mathrm{mmHg}$). Figure 6 shows the convergence rates for each of the two volume meshes. Figure 6 shows that the element quality has a significant impact on the convergence rate of the solution procedure. Indeed, the simulation on the mesh obtained from the STL converges at $1 \cdot \mathrm{e}^{-7}$, whereas the remeshed mesh gives results that are two times more accurate.

DOI: 10.1002/cnm

The next example studies the flow in a simplified aortic arch. The STL triangulation was found on the INRIA web site.[¶] Accurate and converged numerical simulations are mandatory since it has been shown that the flow patterns and the locations of low wall shear stress (WSS) correspond with locations of aneurysm formation in the descending aorta [39, 40]. The WSS is defined as the norm of the shear stress at the wall:

$$\mathrm{WSS} = \|\vec{t}_{\mathrm{w}}\| = \|\vec{t} - ((\vec{t}\cdot\vec{n})\cdot\vec{n})\| \quad \text{with } \vec{t} = \mu(\nabla\vec{u} + \nabla\vec{u}^{\mathrm{T}})\cdot\vec{n}. \tag{7}$$

For the numerical simulation, we apply simple boundary conditions: a parabolic velocity profile at the inlet (heart) and zero natural pressure boundary conditions at the outlets (innominate artery, left common carotid artery, left subclavian artery and descending aorta) and a zero velocity (no-slip) on the vessel walls. We consider a stationary flow at Reynolds $Re = 450$ and different meshes: isotropic volume meshes of, respectively, 28, 160 and 466 k tetrahedra and an adapted anisotropic mesh that has approximately 20k. We fist compute an isotropic surface mesh with our remeshing algorithm and then produce two different types of volume meshes: (i) isotropic volume meshes of different prescribed mesh sizes, (ii) adapted anisotropic volume meshes and (ii) a boundary layer mesh obtained by extrusion of the surface mesh over a number of layers (five layers in the boundary $\delta_{bl} = 1/\sqrt{Re}$). Adaptive refinement in the boundary with either anisotropic metric fields or boundary layers is indeed attractive [41–43] to increase the solution accuracy in the region of interest (at the wall) and this way decrease the load on the solver by reducing the number of finite elements used. With the presented approach of harmonic map, we do have a parametric description of the initial triangulation that enables us to use anisotropic mesh adaptation libraries such as our open-source MadLib library [44]. This library aims at modifying the initial mesh to make it comply with criterions on edge lengths and element shapes by applying a set of standard mesh modifications (edge splits, edge collapses and edge swaps, etc.). An anisotropic field based on the distance to the wall and the curvature can then be defined in order to generate boundary layer meshes. In the example presented in Figure 7(c)), we prescribe a small size with a linear growth in the normal direction to the wall, and three times a larger size is prescribed in the tangent directions. The final mesh metric field is built from those resulting sizes and directions. It should be noted that a volume mesh was also produced from the STL triangulation, but this volume mesh was of too low quality to obtain a convergence of the numerical simulation ($\gamma_{\min} = 1.5\mathrm{e}^{-5}$ and $\bar{\gamma} = 0.45$).

Figure 7 shows the initial STL triangulation, a remeshed isotropic surface mesh, and a mesh cut of the volume anisotropic mesh. As can be seen, initial STL triangulation is faceted and the horizontal structure of the CT slices are visible.

Figure 8 shows the WSS values computed for different meshes at section $A-A'$. We selected section $A-A'$ since this section intersects the regions of low and high WSS. For this section, the WSS values vary in the azimuthal direction, the zero angle corresponding to the location A'. As can be seen in Figure 8, the high-quality isotropic volume meshes converge well toward an azimuthal WSS distribution. The WSS for the anisotropic mesh exhibits more numerical noise that is due to the velocity gradient computations involved in (7) that are less accurate for highly anisotropic meshes [41–43]. Meanwhile, the mean values (max and min WSS) converge toward the one obtained with the finest isotropic mesh within a smaller computational time (mesh of only 20k). The boundary layer volume mesh provides less oscillatory results and shows also convergence toward the finest isotropic mesh for a reduced number of elements (50k versus 1.4M tetrahedra).

The last biomedical computation is the stress computation on a hemipelvis. The initial triangulation (STL file) of the pelvic bone is obtained from a segmentation procedure of a sawbone model that was scanned (CT scan with 1.25 mm thickness). Several isotropic surface meshes are obtained with our automatic remeshing algorithm for different mesh refinements. We analyze the influence of the mesh quality on the accuracy of the solution : five meshes obtained with the uniform remeshing algorithm having, respectively, 420, 270, 70, 20 and 5k triangles, two meshes that are adapted to the curvature with 54 and 8k triangles and three STL triangulations of 10, 5 and

[¶] http://www-c.inria.fr/Eric.Saltel/saltel.php.

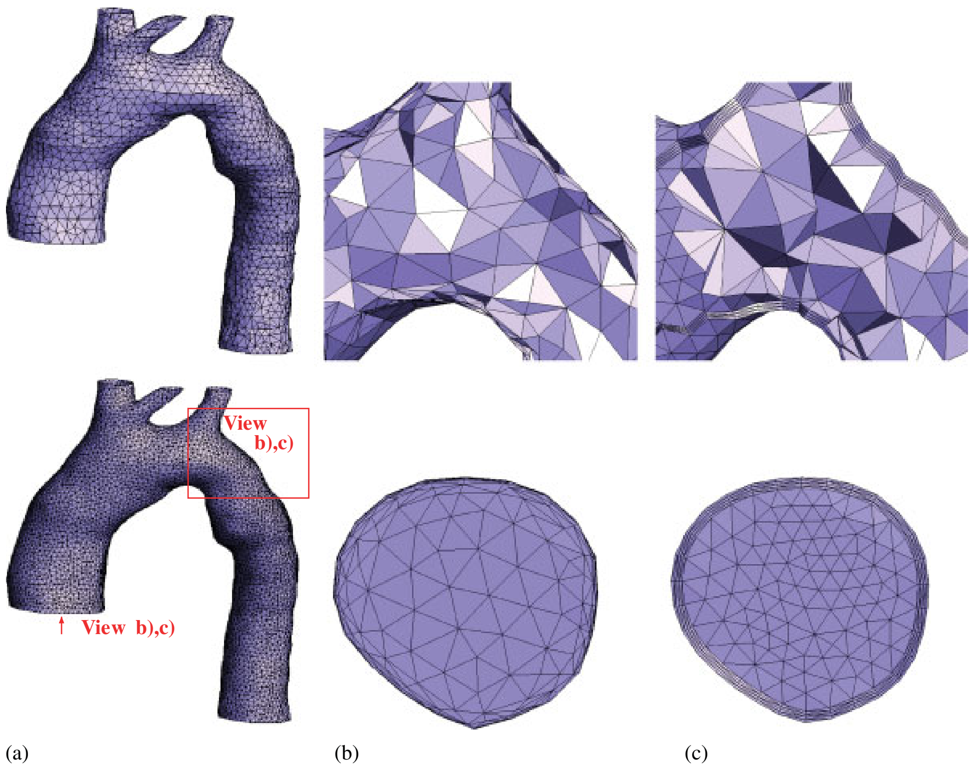


Figure 7. Aortic arch meshes: (a) initial STL triangulation (top) and remeshed surface (isotropic mesh size); (b) anisotropic volume mesh cut created from the remeshed surface with MAdLib; and (c) boundary layer volume mesh.

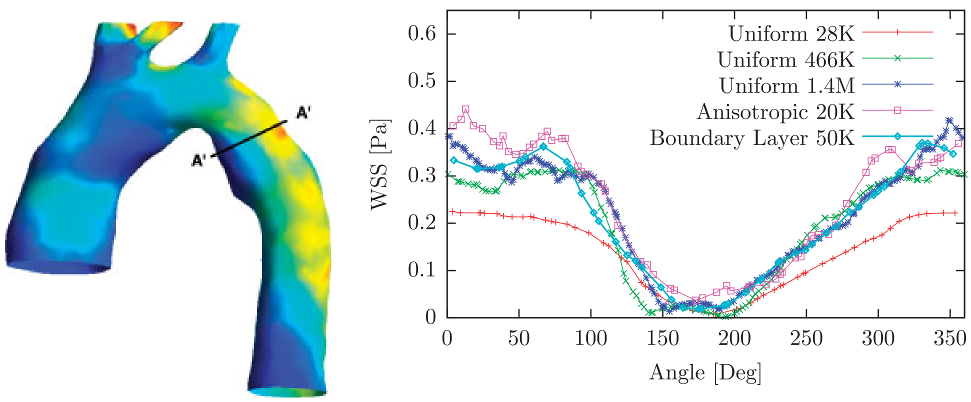


Figure 8. Blood flow simulation in an aortic arch. The left figure shows the WSS distribution and the right figure the WSS along the circumference at section $A-A'$ for different meshes for a constant inlet flow rate. The zero angle corresponds to the location A'.

2k triangles (see Figure 9). The three different STL files are obtained with the meshLab software by refining the triangles or collapsing the edges of the initial STL file of 5k triangles. As expected, the mean quality is $\bar{\kappa}=0.94$ for the remeshed pelvis whereas $\bar{\kappa}=0.66$ for STL triangulations. The curvature adapted meshes are computed by defining the mesh size δ as follows:

$$\delta=\frac{2\pi R}{N_{\mathrm{p}}} \quad \text{with } R=\frac{1}{\kappa} \tag{8}$$

DOI: 10.1002/cnm

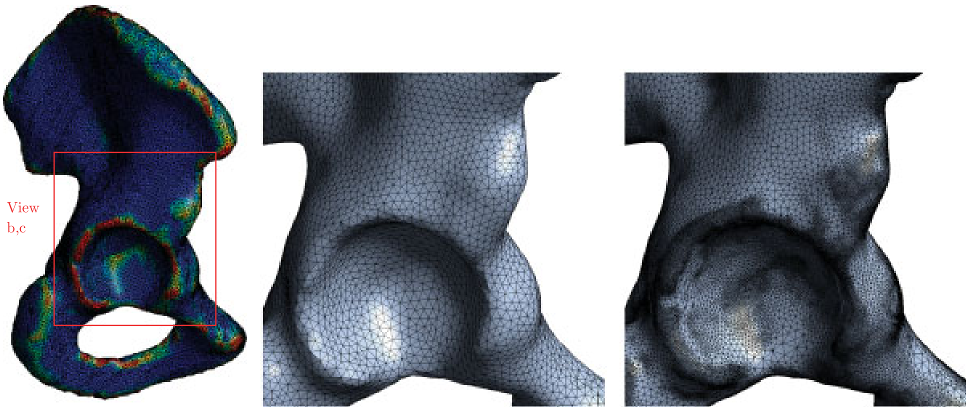


Figure 9. Different meshes used for the mesh convergence analysis: (a) triangulation on which a curvature κ is computed; (b) isotropic remeshed pelvis ($\delta = 0.1$); and (c) curvature-dependent remeshed pelvis (δ given by Equation (8)).

where κ is the curvature that is computed from the initial nodes of the STL triangulation with the algebraic point set surface method (see Figure 9) that is based on the local fitting of algebraic spheres [45] and N_{p} is the number of points chosen for the radius of curvature ($N_{\mathrm{p}} = 15$).

As far as the boundary conditions are concerned, the finite element model is constrained at the sacro-iliac joint and a symmetry boundary condition is applied on the pubic-symphysis. The pelvis is subjected to a 3D load case representative of a single leg stance. Taking a body weight equal to 1000 N, the resulting surface traction force acting on the acetabulum surface is 0.7 MPa. As different meshes are used, the elements forming the boundaries are selected inside a sphere (for the acetabulum and the sacrum) and on one side of a plane (for the pubis) intersecting the pelvis. These fixed boundary conditions are more representative of *in vitro* experimentation than *in vivo* environment but are realistic enough for this analysis [46].

In order to put to the fore the effect of the surface mesh on the behavior of the numerical solution, we analyze the stresses in the cortical bone by using shell elements on the surface of the pelvis. This is well adapted for this analysis because the pelvis has a cortical shell that undergoes most of the stresses. We model this cortical surface layer with a homogeneous shell section of uniform thickness 2 mm, an isotropic Young modulus $E = 18\,000\,\mathrm{N/mm^2}$ and a poisson ratio of $v = 0.3$ [47, 48].

The simulations are computed with the finite element solver Abaqus with linear finite elements. Figure 10 shows the distribution of the Von-Mises stresses developed in the cortical bone with the finest isotropic mesh. The stresses are concentrated around the cotyle and toward the fixed boundary condition at the sacro-iliac joint. The maximum stresses are obtained around the cotyle for the fine meshes, whereas the STL meshes produce higher stresses located above the illium. These are local stress concentrations that appear in small elements, where no significant stress should be present.

To determine the influence of the mesh quality on grid convergence, we consider the different meshes and evaluate the $\mathscr{L}^2$-norm of the discretization error on a given h-mesh:

$$\|e\|^2_{\mathscr{L}^2} = \int_{\Omega} \sum_{j=1}^{3} (u^h_j - \bar{u}_j)^2 \,\mathrm{d}\Omega \tag{9}$$

where the subscript j denotes the jth component of the displacement vector and $\bar{u}$ is the interpolation of the displacement vector obtained with the finest mesh of 420k elements on the considered h-mesh.

Figure 10 shows the grid convergence for the different meshes, where we can clearly see the influence of the mesh quality on the convergence. The theoretical convergence for linear shell

DOI: 10.1002/cnm

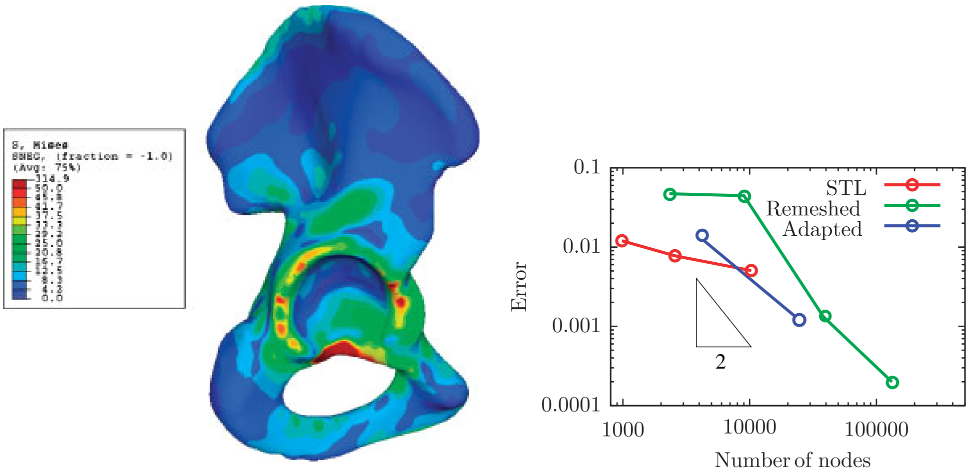


Figure 10. Von mises stress distribution in the cortical shell computed with the finest isotropic surface mesh (Left) and influence of the mesh quality on the numerical convergence (Right). We show the $\mathscr{L}^2$-norm of the discretization error as a function of the number of nodes for different meshes.

elements of order $\mathscr{O}(h^2)$ is reached for our high-quality meshes, whereas the STL triangulations present a much lower convergence $\mathscr{O}(h^{0.2})$ as well as a higher error for the same number of nodes. The curvature adapted meshes are attractive since for the same discretization error, they present less nodes. However, an adaptive mesh with a mesh size δ given by *a posteriori* an error-estimator would be better then a curvature-based adaptive mesh.

5. CONCLUSION

In this work, we have presented a fully automatic approach to recover a high-quality surface mesh from low-quality oversampled inputs (STL files) obtained via 3D acquisition systems. The approach is original as it combines an efficient and robust parametrization technique based on harmonic maps [17] with a multi-level edge partitioning algorithm that partitions the mesh into a small number of partitions. With the present approach, we are able to remesh any surface with any topological genus and with large geometrical aspect ratio such as arteries. We showed that the remeshing procedure is highly efficient and produces high-quality meshes that are suitable for finite element biomedical simulations. We have presented several biomedical computations that quantify the influence of the mesh quality on the convergence behavior.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the Belgian National Fund for Scientific Research (FNRS) and the Walloon Region (EFCONIVO Project) for their financial support.

REFERENCES

1. Gross M. Computer graphics in medicine: from visualization to surgery simulation. *ACM SIGGRAPH Computer Graphics* 1998; **32**(1):53–56.
2. Schoberl J. Netgen an advancing front 2d/3d-mesh generator based on abstract rules. *Computing and Visualization in Science* 1997; **1**:41–52.
3. Si H. Tetgen a quality tetrahedral mesh generator and three-dimensional delaunay triangulator. 2004. Available from: http://tetgen.berlios.de/.
4. Szczerba D, McGregor R, Szekely G. High quality surface mesh generation for multi-physics bio-medical simulations. *Computational Science—ICCS 2007*, vol. 4487. Springer: Berlin, 2007; 906–913.

5. Batdorf M, Freitag L, Ollivier-Gooch C. A computational study of the effect of unstructured mesh quality on solution efficiency. Presented at the *13th Annual Computational Fluid Dynamics Meeting*, Snowmass Village, 1997.
6. Babuska I, Aziz A. On the angle condition in the finite element method. *SIAM Journal on Numerical Analysis* 1976; **13**:214–226.
7. Fried I. Condition of finite element matrices generated from nonuniform meshes. *AIAA Journal* 1972; **10**:219–221.
8. Ito Y, Nakahashi K. Direct surface triangulation using stereolithography data. *AIAA Journal* 2002; **40**(3):490–496.
9. Bechet E, Cuilliere JC, Trochu F. Generation of a finite element mesh from stereolithography (stl) files. *Computer-Aided Design* 2002; **34**(1):1–17.
10. Wang D, Hassan O, Morgan K, Weatheril N. Enhanced remeshing from stl files with applications to surface grid generation. *Communications in Numerical Methods in Engineering* 2007; **23**:227–239.
11. Borouchaki H, Laug P, George P. Parametric surface meshing using a combined advancing-front generalized delaunay approach. *International Journal for Numerical Methods in Engineering* 2000; **49**:223–259.
12. Zheng Y, Weatherill N, Hassan O. Topology abstraction of surface models for three-dimensional grid generation. *Engineering with Computers* 2001; **17**:28–38.
13. Marcum DL. Efficient generation of high-quality unstructured surface and volume grids. *Engineering with Computers* 2001; **17**:211–233.
14. Tristano J, Owen S, Canann S. Advancing front surface mesh generation in parametric space using Riemannian surface definition. *Proceedings of Seventh International Meshing Roundtable*, Sandia National Laboratory, Dearborn, MI, 1998; 429–455.
15. Laug P, Boruchaki H. Interpolating and meshing 3d surface grids. *International Journal for Numerical Methods in Engineering* 2003; **58**:209–225.
16. Levy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. *Computer Graphics* (*Proceedings of SIGGRAPH 02*), San-Antonio, TX, U.S.A. ACM: New York, 2002; 362–371.
17. Remacle JF, Geuzaine C, Compère G, Marchandise E. High quality surface remeshing using harmonic maps. *International Journal for Numerical Methods in Engineering* 2009; DOI: 10.1002/nme.2824.
18. Floater MS, Hormann K. Surface parameterization: a tutorial and survey. In *Advances in Multiresolution for Geometric Modelling*, Dodgson NA, Floater MS, Sabin MA (eds). Springer: Berlin, 2005; 157–186.
19. Sheffer A, Praun E, Rose K. Mesh parameterization methods and their applications. *Foundations and Trends in Computer Graphics and Vision* 2006; **2**(2):105–171.
20. Marcum DL, Gaither A. Unstructured surface grid generation using global mapping and physical space approximation. *Proceedings of Eighth International Meshing Roundtable*, South Lake Tahoe, CA, U.S.A., October 1999; 397–406. DOI: 10.1007/PL00013386.
21. Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W. Multiresolution analysis of arbitrary meshes. *SIGGRAPH '95*: *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, Los Angeles, CA, U.S.A., 1995; 173–182.
22. Geuzaine C, Remacle JF. Gmsh: a three-dimensional finite element mesh generator with built-in pre-post-processing, facilities. *International Journal for Numerical Methods in Engineering* 2009; **79**(11):1309–1331.
23. Shinagawa Y, Kunii T, Kergosien YL. Surface coding based on morse theory. *IEEE Computer Graphics and Applications* 1991; **11**(5):66–78.
24. Hendrickson B, Leland R. A multilevel algorithm for partitioning graphs. *Proceedings on Supercomputing*, Albuquerque, 1995.
25. Karypis G, Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Computing* 1999; **20**(1):359–392.
26. Chang CT, Gorissen B, Melchior S. Fast computation of the minimportanceimal oriented bounding box on the rotation group so(3). *ACM Transactions on Graphics* 2009; in press.
27. Durand N, Alliot JM. A combined nelder-mead simplex and genetic algorithm. *GECCO'99*: *Proceedings of Genetic and Evolutionary Computation Conference*, San Francisco, 1999; 1–7.
28. Frey PJ, George PL. *Mesh Generation*, *Application to Finite Elements*. Hermes Science: Europe, 2000.
29. Ethier CR, Steinman DA, Zhang X, Karpik SR, Ojha M. Flow waveform effects on end-to-side anastomotic flow patterns. *Journal of Biomechanics* 1998; **31**(7):609–617.
30. Sherwin SJ, Shah O, Doorly DJ, Peiro J, Papaharilaou Y, Watkins N, Caro CG, Dumoulin CL. The influence of out-of-plane geometry on the flow within a distal end-to-side anastomosis. *Journal of Biomechanical Engineering* 2000; **122**:1–10.
31. Sherwin SJ, Doorly DJ. In *Vascular Grafts*: *Experiment and Modelling*, Chapter 9, Tura A (ed.). WIT Press, 2003; 327–373.
32. Giordana S, Sherwin SJ, Peiro J, Doorly DJ, Crane JS, Lee K, Cheshire NJW, Caro CG. Local and global geometric influence on steady flow in distal anastomoses of peripheral by-pass grafts. *Journal of Biomechanical Engineering* 2005; **127**(7):1087–1098.
33. Longest PW, Vinchurkar S. Effects of mesh style and grid convergence on particle deposition in bifurcating airway models with comparisons to experimental data. *Medical Engineering and Physics* 2007; **29**:350–366.
34. Vinchurkar S, Longest PW. Evaluation of hexahedral, prismatic and hybrid mesh styles for simulating respiratory aerosol dynamics. *Computers and Fluids* 2008; **37**:317–331.

35. Liu Y, Pekkan K, Jones SC, Yoganathan AP. The effects of different mesh generation methods on computational fluid dynamic analysis and power loss assessment in total cavopulmonary connection. *Journal of Biomechanical Engineering* 2004; **126**(5):594–604.
36. Prakash S, Ethier CR. Requirements for mesh resolution in 3d computational hemodynamics. *Journal of Biomechanical Engineering* 2001; **123**(2):134–144.
37. Marchandise E, Remacle JF. A stabilized finite element method using a discontinuous level set approach for solving two phase incompressible flows. *Journal of Computational Physics* 2006; **219**:780–800. DOI 10.1016/j.jcp.2006.04.015.
38. Ethier CR, Prakash S, Steinman DA, Leask RL, Couch GG, Ojha M. Steady flow separation patterns in a 45 degree junction. *Journal of Fluid Mechanics* 2000; **41**:1–38.
39. O'brien KD, Chait A. The biology of the artery wall in atherogenesis. *The Medical Clinics of North America* 1994; **78**(1):41–67.
40. Shaaban AM, Duerinckx A. Wall shear stress and early atherosclerosis. *American Journal of Roentgenology* 2000; **174**:1657–1665.
41. Sahni O, Mueller J, Jansen K, Shephard M, Taylor C. Efficient anisotropic adaptive discretization of cardiovascular system. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**:5634–5655.
42. Sahni O, Jansen K, Shephard M. Automated adaptive cardiovascular flow simulations. *Engineering with Computers* 2009; **25**(1):25–36.
43. Botti L, Piccinelli M, Ene-Iordache B, Remuzzi A, Antiga L. An adaptive mesh refinement solver for large-scale simulation of biological flows. *Communications in Numerical Methods in Engineering* 2009; **26**(1):86–100. Special Issue: Patient-Specific Computational Modelling.
44. Compère G, Remacle JF. Website of madlib: mesh adaptation library 2008. Available from: http://www.madlib.be.
45. Guennebaud G, Germann M, Gross M. Dynamic sampling and rendering of algebraic point set surfaces. *Computer Graphics Forum* 2008; **27**:653–662.
46. Phillips A, Pankaj P, Howie C, Usmani A, Simpson A. Finite element modelling of the pelvis: inclusion of muscular and ligamentous boundary conditions. *Medical Engineering and Physics* 2007; **29**(7):739–748.
47. Dalstra M, Huiskes R. Load transfer across the pelvic bone. *Journal of Biomechanics* 1995; **28**:715–724.
48. Anderson A, Peters C, Tuttle B, Weiss J. Subject-specific finite element model of the pelvis: development, validation and sensitivity studies. *Journal of Biomechanical Engineering* 2005; **127**(3):364–373.

DOI: 10.1002/cnm

Appendix D

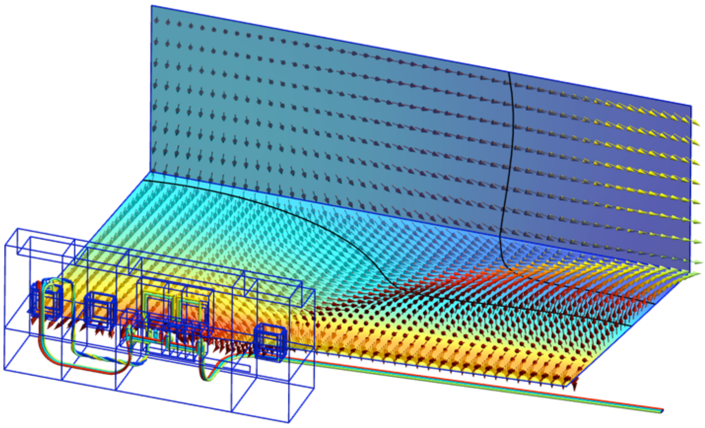
Abstract Post-Processing Interface

The post-processing module can load, transform and display multiple post-processing datasets (called "views") at once, along with the geometry and the mesh. Each view can contain a mix of scalar, vector and tensor data as well as text annotations. Views can be manipulated either globally or individually (each view has its own button in the GUI and can be referred to by its index in a script), and each one possesses its own set of display options. Internally, the view is an abstract class that can access a variety of underlying representations, from the node-based data sets used in standard finite element codes to high-order, discontinuous data sets used, e.g., in discontinuous Galerkin or finite volume solvers [24].

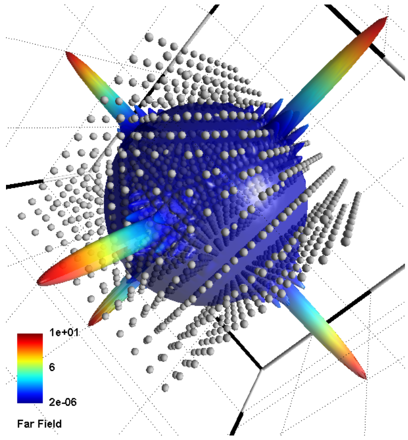
Scalar fields are represented by iso-surfaces or color maps, while vector fields are represented by three-dimensional arrows or displacement maps (see Figure D.1). The graphics display code is written using OpenGL, and all representations are stored internally as vertex arrays to improve rendering performance.

Display options are non-destructive (they do not modify the dataset) and can be changed on the fly. These options include for example choosing the plot type and the number of iso-surfaces to display, modifying the type and range of the scale and the colormap, or applying complex geometrical transformations—changes of coordinates based on functional expressions, e.g. to exploit symmetry or to apply an geometrical offset based on the values in the dataset. In addition to the non-destructive display options, the post-processing module provides a plug-in architecture to enable the application of destructive modifications to views. Gmsh ships with about thirty default plug-ins, that perform operations such as computing sections, elevation maps and stream lines, extracting boundaries, components and time steps, applying differential operators, calculating eigenvalues and eigenvectors, or triangulating point datasets.

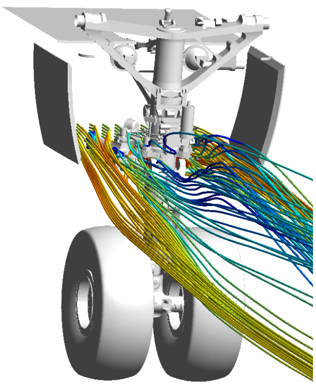
All the post-processing features can be accessed either interactively or through the scripting language, which permits to automate all operations, as for example to create animations. Gmsh provides a large number of raster output formats, as well as vector output for high-quality technical renderings using GL2PS [10], which is especially useful for 2-D scenes. Raster files can also be created at sizes larger than what the screen resolution allows by using offscreen rendering [19].



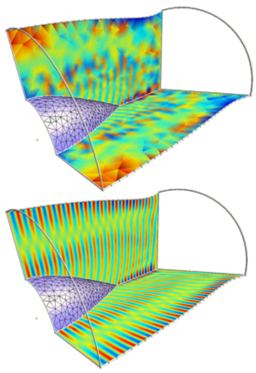
cuts, iso-curves and vectors [2]
(K. A. Berger & B. Kubicek, Arsenal Research)



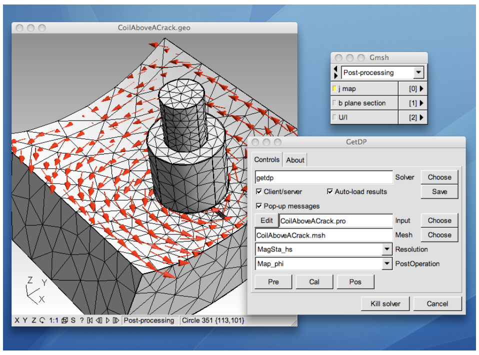
elevation map



streamlines
(P. Geuzaine, Cenaero)



high-order
(Taken from Reference [24])



solver interface

Figure D.1: Some images from the solver and post-processing modules.

Bibliography

[1] E. Bechet, J.-C. Cuillere, and F. Trochu. Generation of a finite element mesh from stereolithography (stl) files. *Computer-Aided Design*, 34(1):1–17, 2002.

[2] K. A. Berger and B. Kubicek. Magnetic field of a 30kV/400V-substation, 2008. Private communication, Arsenal Research, Austria.

[3] J.-C. Cuilliere. An adaptive method for the automatic triangulation of 3d parametric surfaces. *Computer-Aided Design*, 2:139–149, 1998.

[4] R. A. Dwyer. A simple divide-and-conquer algorithm for computing delaunay triangulations in o(n log log n) expected time. In *Proceedings of the second annual symposium on Computational geometry*, pages 276 – 284, 1986.

[5] L. A. Freitag and C. Ollivier-Gooch. Tetrahedral mesh improvement using face swapping and smoothing. *International Journal for Numerical Methods in Engineering*, 40(21):3979–4002, 1998.

[6] P. J. Frey. About surface remeshing. In *9th International Meshing Roundtable*, 2000.

[7] P.-L. George and H. Borouchaki. *Delaunay Triangulation and Meshing: Application to Finite Elements*. Hermes, 1998.

[8] P.-L. George and P. Frey. *Mesh Generation*. Hermes, 2000.

[9] P.-L. George, F. Hecht, and E. Saltel. Automatic mesh generator with specified boundary. *Computer Methods in Applied Mechanics and Engineering*, 92(3):269–288, 1991.

[10] C. Geuzaine. GL2PS: an OpenGL to PostScript printing library, 2000. `http://www.geuz.org/gl2ps/`.

[11] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. 79(11):1309–1331, 2009.

[12] R. Haimes. CAPRI: Computational analysis programming interface (a solid modeling based infra-structure for engineering analysis and design). Technical report, Massachusetts Institute of Technology, 2000.

[13] F. Hecht. Bamg: Bidimensional anisotropic mesh generator, 2006. `http://www.freefem.org/ff++`.

[14] P. Laug and H. Borouchaki. Blsurf-mesh generator for composite parametric surfaces-user's manual, 1999. Technical Report, INRIA, France.

[15] X. Li. *Mesh Modification procedure for General 3-D Non-Manifold Domains.* PhD thesis, Renselear Polytechnic Indtitute, 2003.

[16] X. Li, J.-F. Remacle, N. Chevaugeon, and M. S. Shephard. Anisotropic mesh gradation control. In *13th International Meshing Roundtable*, 2004.

[17] A. Liu and B. Joe. Relationship between tetrahedron shape measures. *BIT Numerical Mathematics*, 34(2):268–287, 1994.

[18] H. Lo and W. X. Wang. Generation of anisotropic mesh by ellipse packing over an unbounded domain. *Engineering with Computers*, 20(4):372–383, 2005.

[19] B. Paul. The Mesa 3D graphics library, 1995. `http://www.mesa3d.org/`.

[20] P. P. Pebay and T. J. Baker. Analysis of triangle quality measures. *Mathematics of Computation*, 72(244):1817–1839, 2003.

[21] L. Piegl and W. Tiller. *The Nurbs Book.* Springer, 1997.

[22] S. Rebay. Efficient unstructured mesh generation by means of delaunay triangulation and bowyer-watson algorithm. *Journal of Computational Physics*, 106:25–138, 1993.

[23] J.-F. Remacle, X. Li, M. S. Shephard, and J. E. Flaherty. Anisotropic adaptive simulation of transient flows using discontinuous galerkin methods. *International Journal for Numerical Methods in Engineering*, 62(7):899–923, 2005.

[24] J.-F. Remacle, E. Marchandise, N. Chevaugeon, and C. Geuzaine. Efficient visualization of high order finite elements. *International Journal for Numerical Methods in Engineering*, 69(4):750–771, 2007.

[25] J. Schöberl. Netgen, an advancing front 2d/3d-mesh generator based on abstract rules. *Comput. Visual. Sci.*, 1:41–52, 1997.

[26] M. Segal and K. Akeley. The OpenGL graphics system: A specification. Technical report, Silicon Graphics Computer Systems, 1992.

[27] J. R. Shewchuk. Robust Adaptive Floating-Point Geometric Predicates, May 1996.

[28] K. Shimada, A. Yamada, and T. Itoh. Anisotropic triangular meshing of parametric surfaces via close packing of ellipsoidal bubbles. In *6th International Meshing Roundtable*, pages 375–390,, 1997.

[29] H. Si. Tetgen a quality tetrahedral mesh generator and three-dimensional delaunay triangulator, 2004. `http://tetgen.berlios.de/`.

[30] D. F. Watson. Computing the n-dimensional delaunay tessellation with application to voronoï polytopes. *the Computer Journal*, 24(2):167–175, 1981.

[31] N. P. Weatherill. The integrity of geometrical boundaries in the two-dimensional delaunay triangulation. *Communications in Applied Numerical Methods*, 6(2):101–109, 1990.

[32] M. A. Yerry and M. S. Shephard. Automatic three-dimensional mesh generation by the modified-octree technique. *International Journal for Numerical Methods in Engineering*, 20(11):1965–1990, 1984.