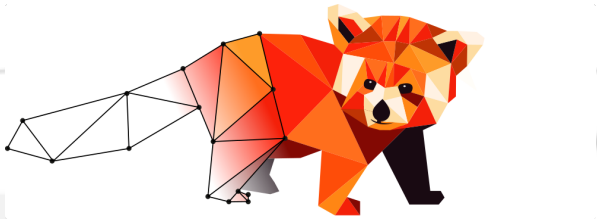
Mmg platform : robust, open source and multidisciplinary software for remeshing.

C. Dapogny, C. Dobrzynski,
P. Frey, A. Froehly

October, 19th



upgrade your meshes

Introduction

Surface remeshing

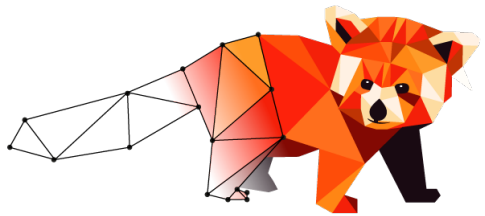
Mesh improvement/adaptation

Level-set discretization

Mmg platform in action

open source community

Introduction

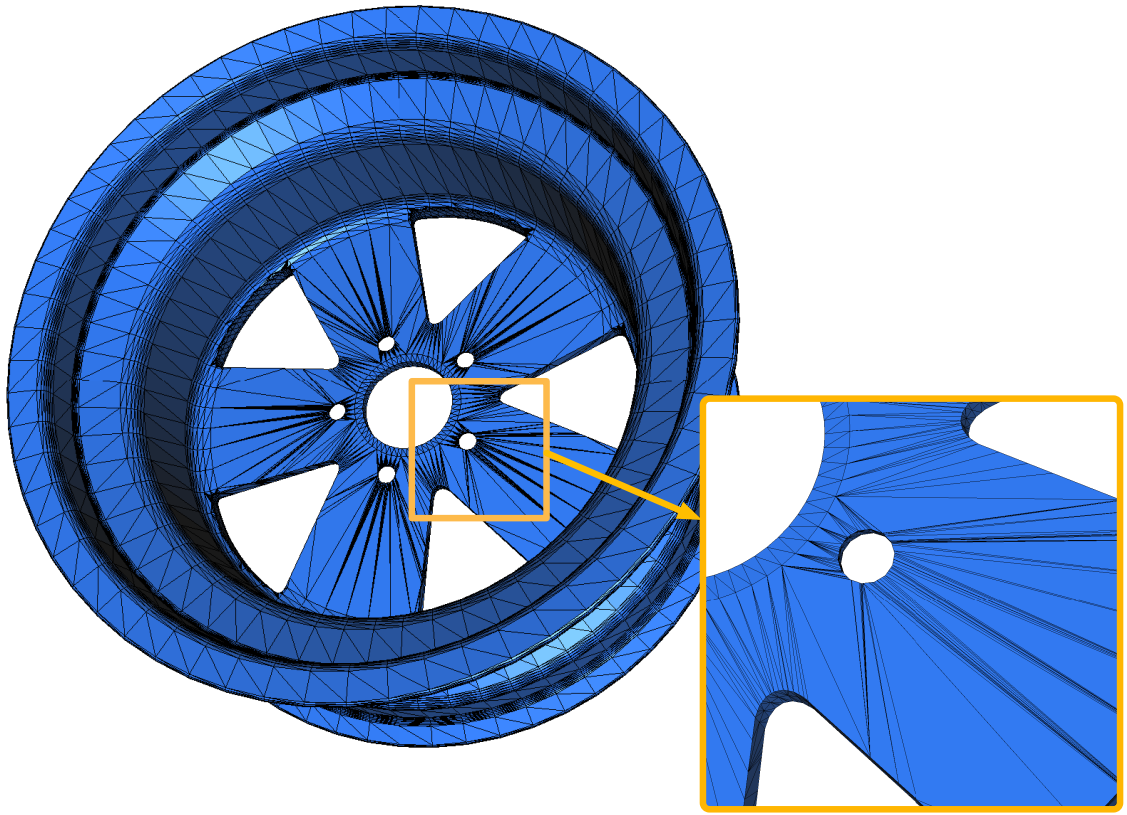


Upgrade
your meshes

Why remeshing ?

Multiple reasons may impose to modify a mesh.

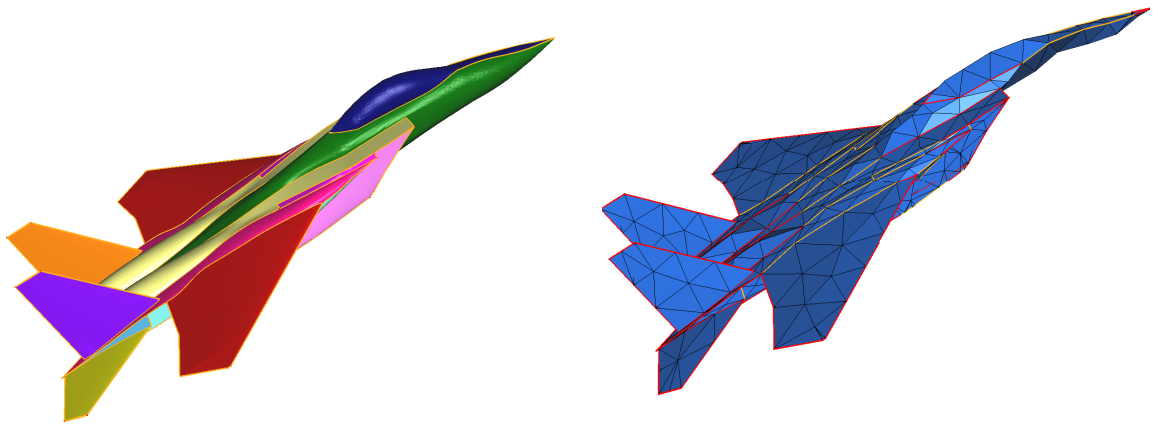
1. The input mesh contains elements with very bad quality



A surface mesh containing almost degenerate triangles

Why remeshing ?

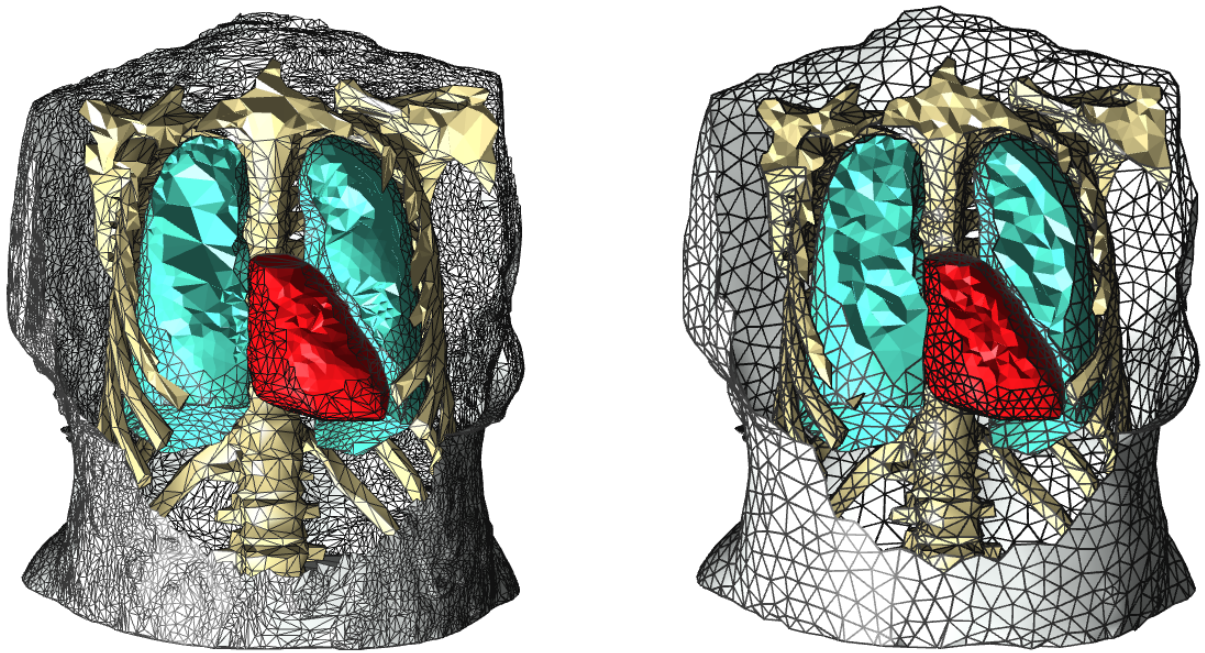
2. The input mesh is a poor geometric approximation of the domain it is meant to represent



A mesh which poorly approximates the surface it is intended for

Why remeshing ?

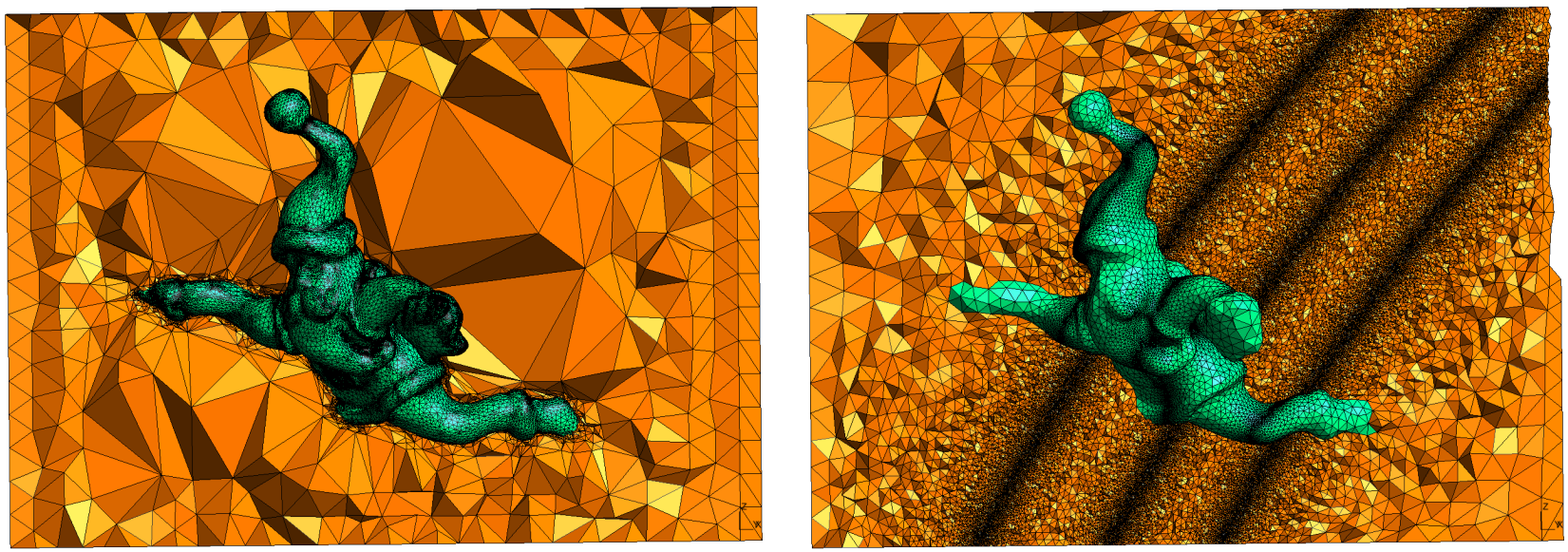
3. The input mesh contains too many elements for what we needed to do.



CT-scan segmentation (left) and computational mesh (right)
(N. Zemzemi, Carmen team)

Why remeshing ?

4. The sizes of the elements in the mesh should respect a user-defined size map

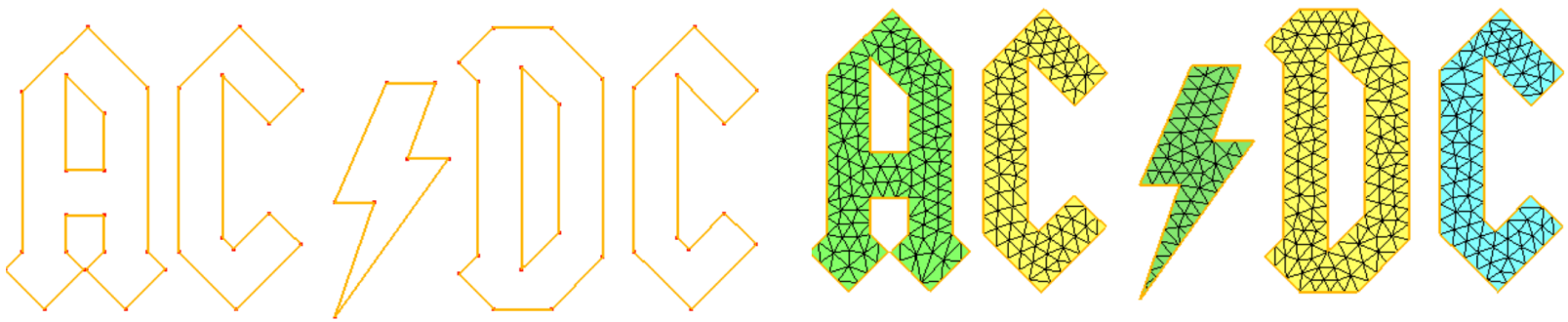


Init mesh (left) and adaptation of this mesh (right)

Mmg platform

Features

1. 2D Mesh generation ;
2. Mesh improvement ;
3. Isotropic/anisotropic mesh adaptation ;
4. Level-Set discretization/mesh of an implicit domain.

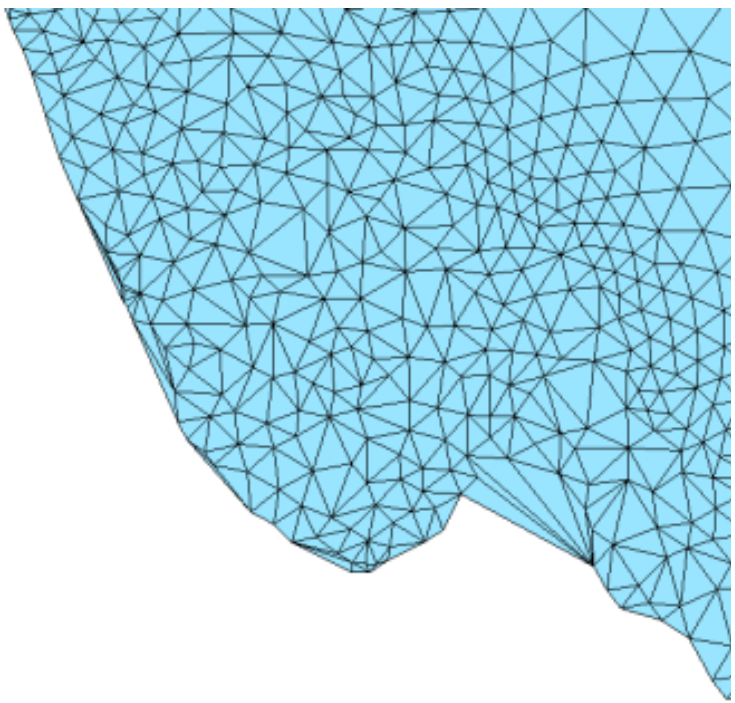


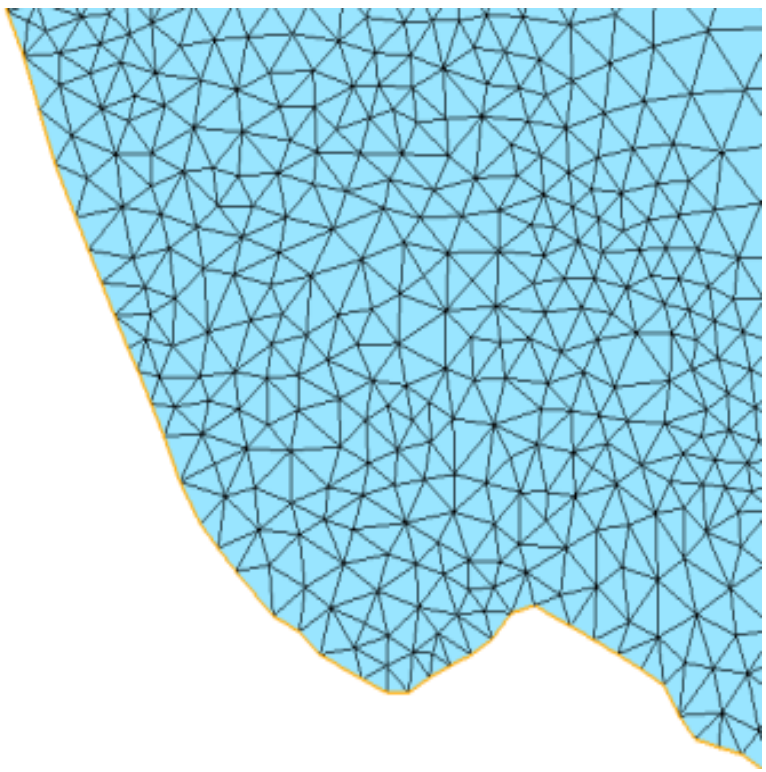
upgrade your meshes

Mmg platform

Features

1. 2D Mesh generation ;
2. Mesh improvement ;
3. Isotropic/anisotropic mesh adaptation ;
4. Level-Set discretization/mesh of an implicit domain.





Mmg platform

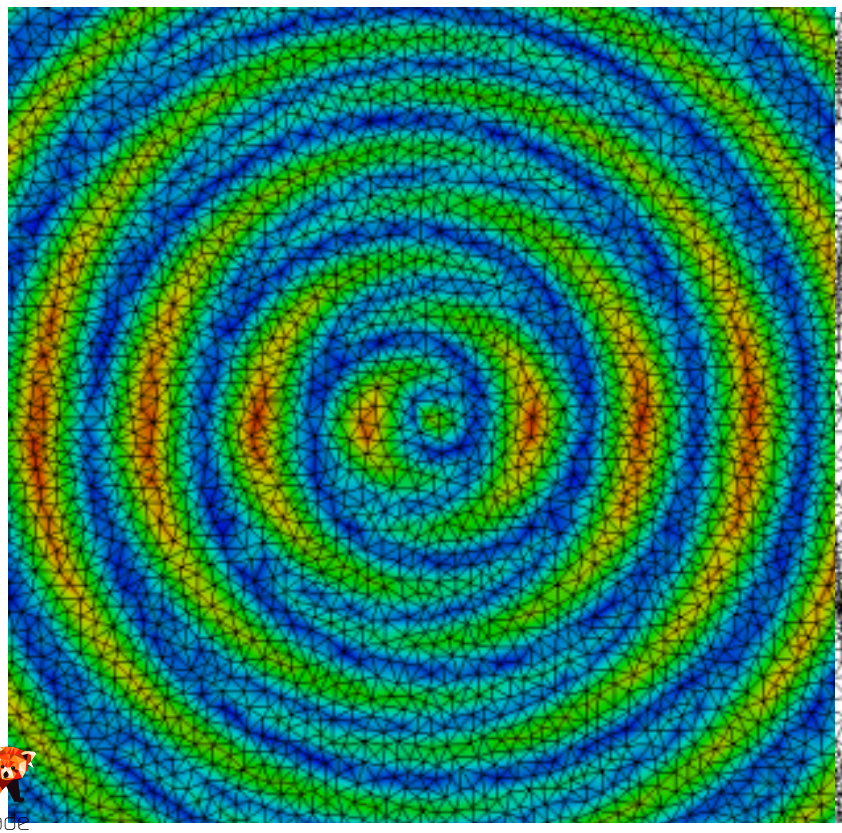
Features

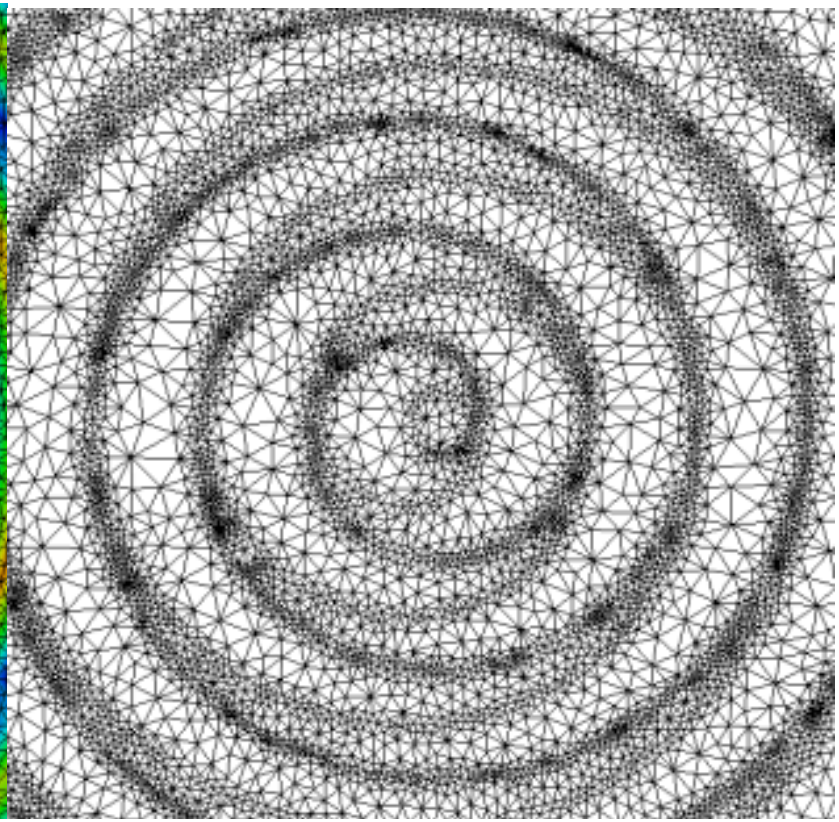
1. 2D Mesh generation ;
2. Mesh improvement ;
3. Isotropic/anisotropic mesh adaptation ;
4. Level-Set discretization/mesh of an implicit domain.

Mmg platform

Features

1. 2D Mesh generation ;
2. Mesh improvement ;
3. Isotropic/anisotropic mesh adaptation ;
4. Level-Set discretization/mesh of an implicit domain.

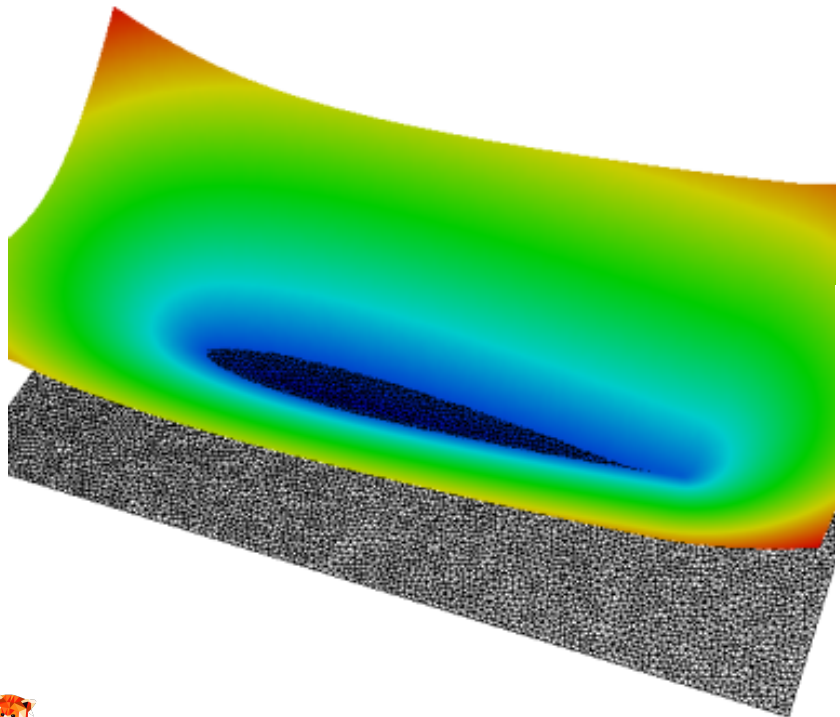


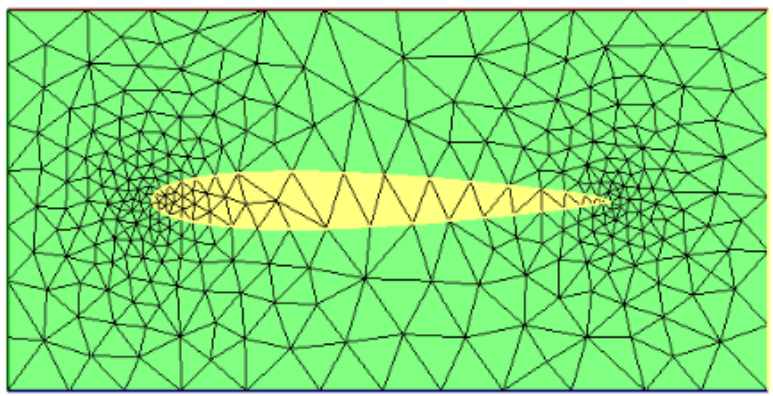


Mmg platform

Features

1. 2D Mesh generation ;
2. Mesh improvement ;
3. Isotropic/anisotropic mesh adaptation ;
4. Level-Set discretization/mesh of an implicit domain.





Mmg platform

Features

1. 2D Mesh generation ;
2. Mesh improvement ;
3. Isotropic/anisotropic mesh adaptation ;
4. Level-Set discretization/mesh of an implicit domain.

Three software

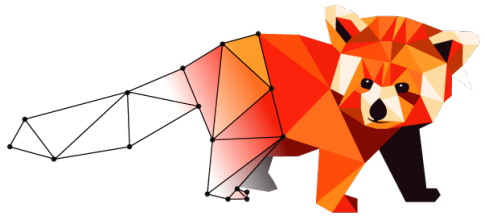
1. Mmg2d : two-dimensional mesher/remesher and isovalue discretization
2. Mmgs : surface remesher and isovalue discretization
3. Mmg3d : three-dimensional remesher and isovalue discretization

Visit our webpage :

`www.mmgtools.org`

upgrade your meshes

Surface remeshing



upgrade
your meshes

Challenges and main algorithm

Requirements

1. Properly detect the singularities of the geometry
2. Forbid the geometry degeneracy
3. Keep a sufficiently good approximation of the geometry
4. Gives to the users a control of the geometric approximation

Goals

1. Have a good surface satisfying the user requirements with a minimum of points
2. Improve the surface mesh quality

Algorithm

1. Singularities detection
2. Ideal underlying geometry construction
3. Surface improvement w.r. to the Hausdorff distance

Singularities detection

Ridges/geometric edges

edges delimiting two portions of surface which intersect a sharp angle

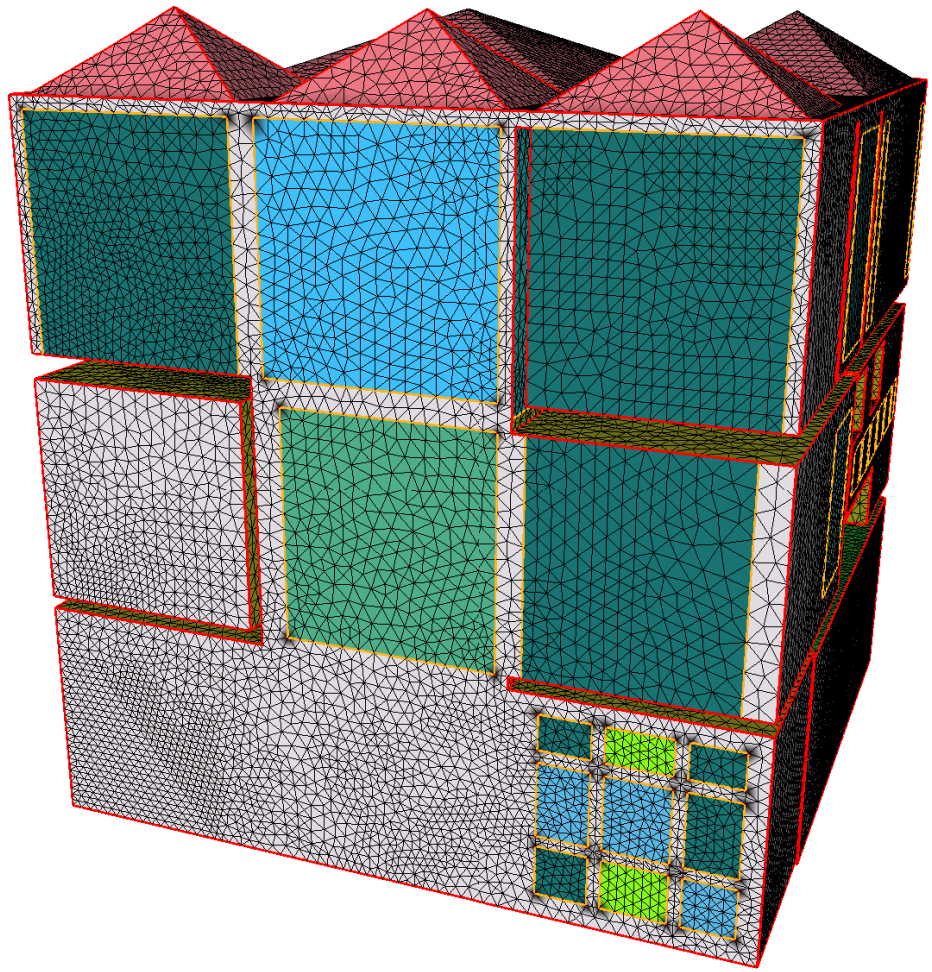
Reference edges

edges at the interface of two different sub-domains

Corners

points which arise as endpoints of at least 3 special edges

Required entities

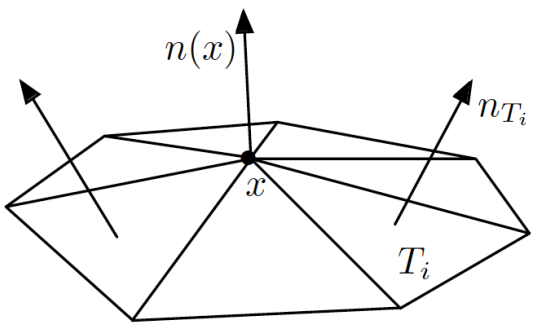


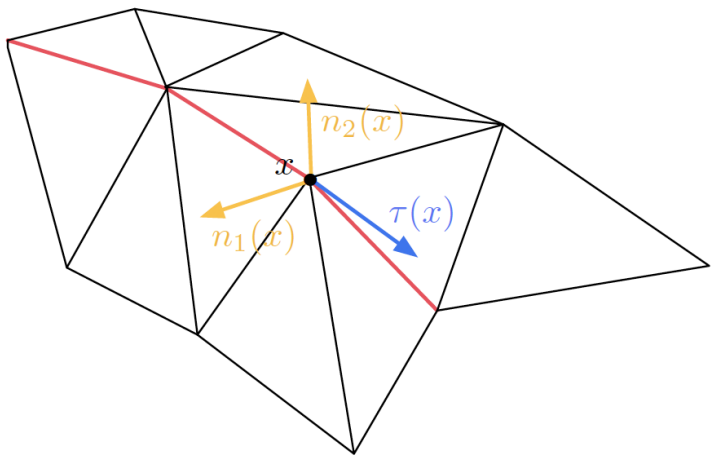
Underlying geometry

Normal computation at vertex P

$$n(P) \approx \frac{\sum_{Tr \ni P} \alpha_{Tr} n_{Tr}}{|\sum_{Tr \ni P} \alpha_{Tr} n_{Tr}|}$$

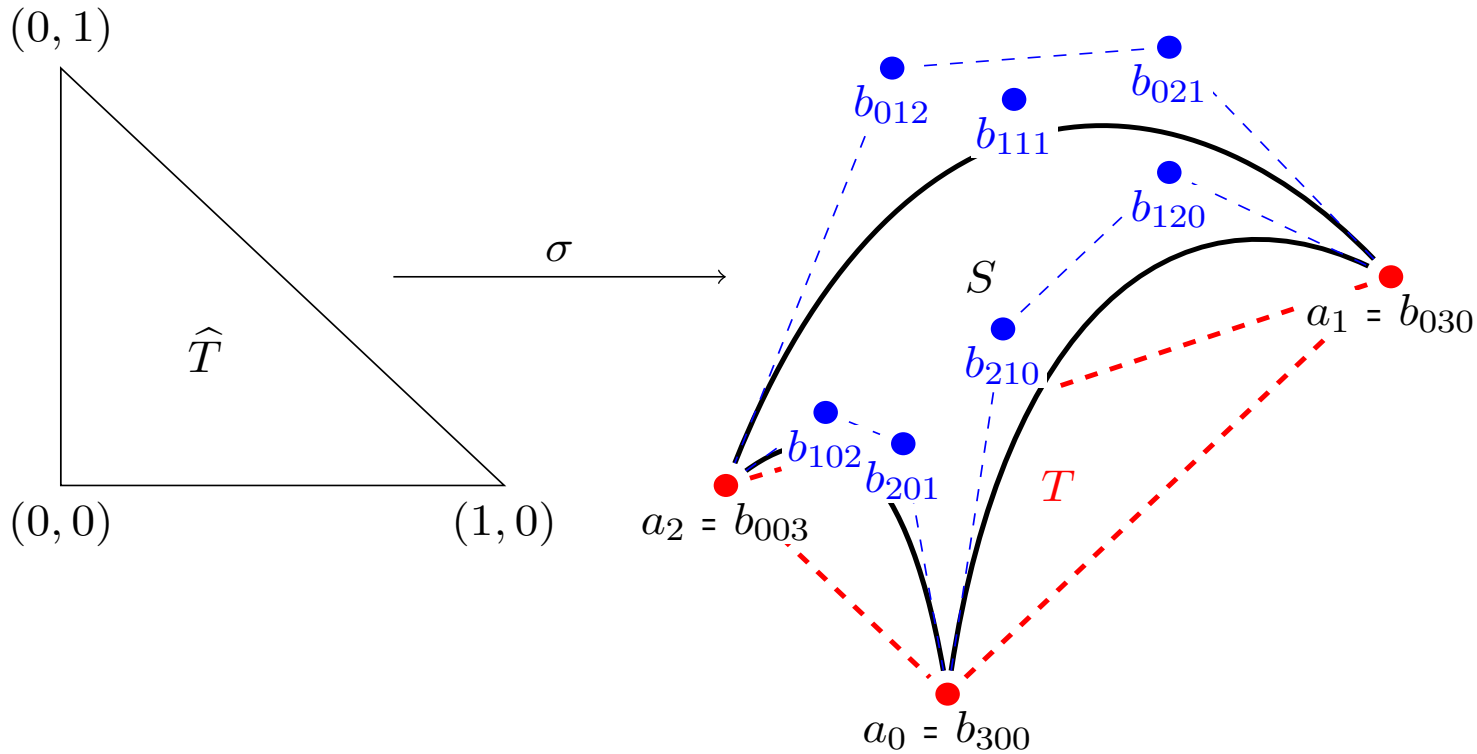
$\alpha_{Tr} \in [0, 1]$ such that $\sum_{Tr \ni P} \alpha_{Tr} = 1$
α_{Tr} proportional to the area of Tr
n_{Tr} the unit normal to the triangle Tr





Normal computation at a regular vertex and a ridge one

Underlying geometry

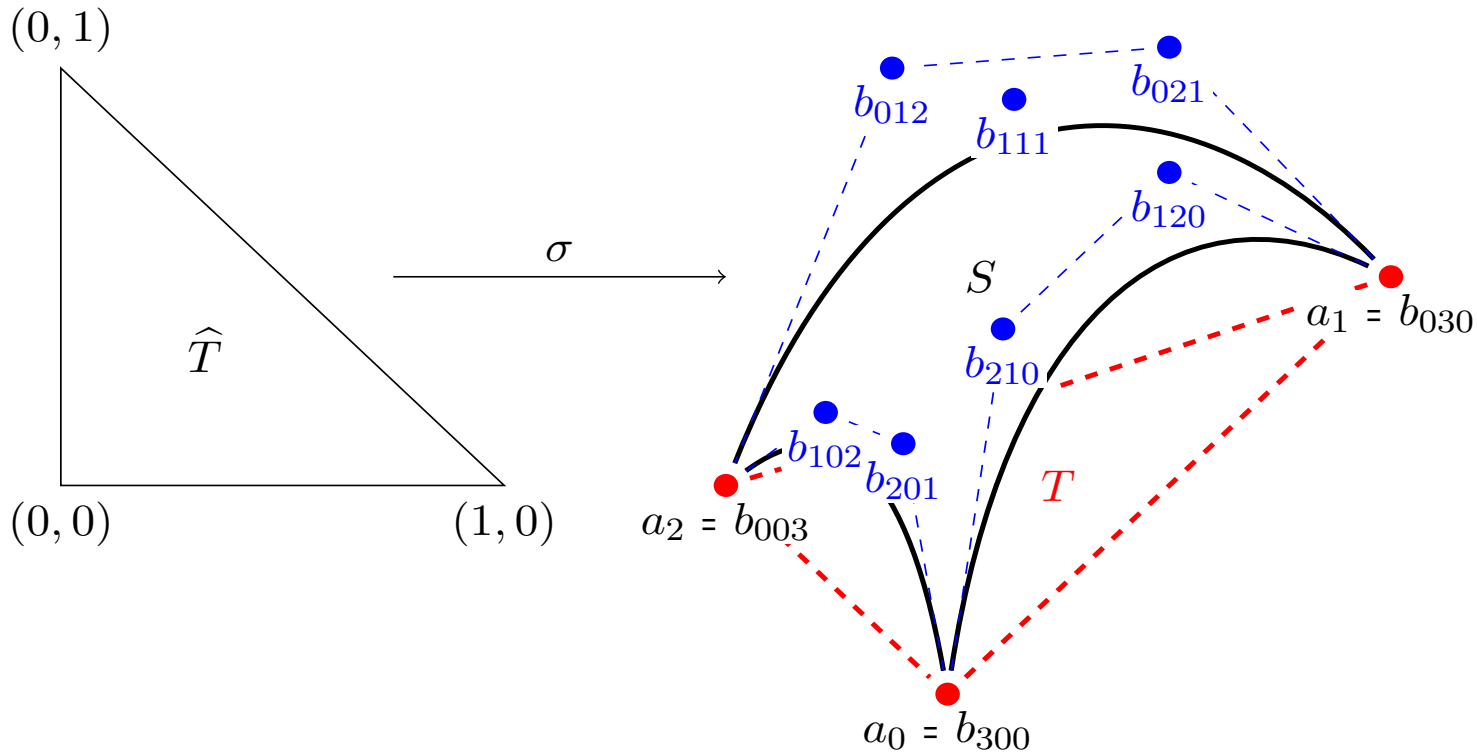


Cubic piece of surface (Bézier cubic polynomial)

$\forall (u, v) \in \hat{T},$

$$\sigma(u, v) = \sum_{i,j \in 0..3} \frac{3!}{i!j!k!} (1 - u - v)^i u^j v^{1-i-j} b_{i,j,k}$$

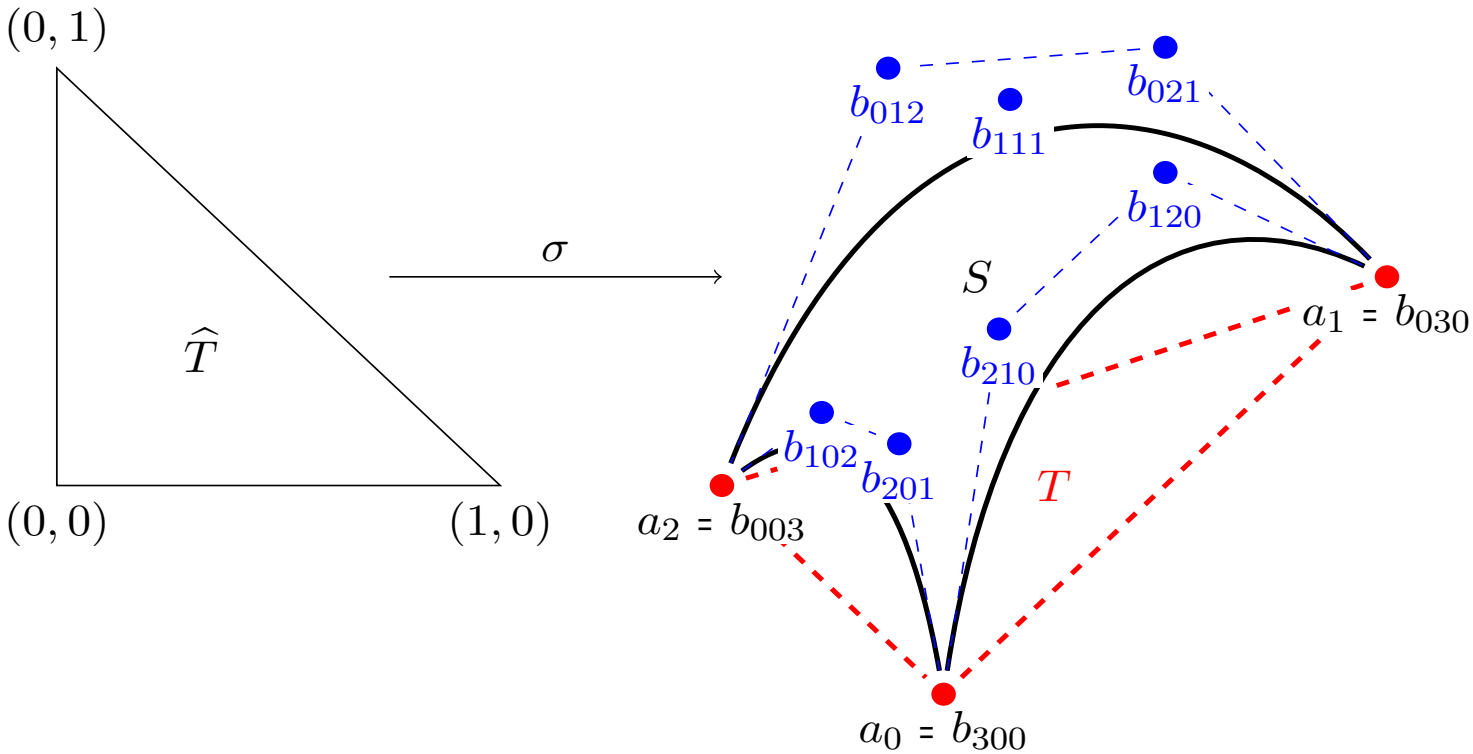
Underlying geometry



Choice of the edges control points

- curved edges independant of a_i
- tangents at $a_i \perp n(a_i)$
- tangents at a_i non colinear
- curved edges are geodesic curves

Underlying geometry



Choice of the middle control point $b_{1,1,1}$

- if there exists a quadratic polynomial parametrization with the same edges then the two parametrizations are equal.

Control the surface approximation

Let K_1, K_2 be two compact subsets of $\mathbb{R}^d$.

Euclidian distance

$$\text{For any } x \in \mathbb{R}^d, \qquad d(x, K_1) = \inf_{y \in K_1} d(x, y)$$

Hausdorff distance

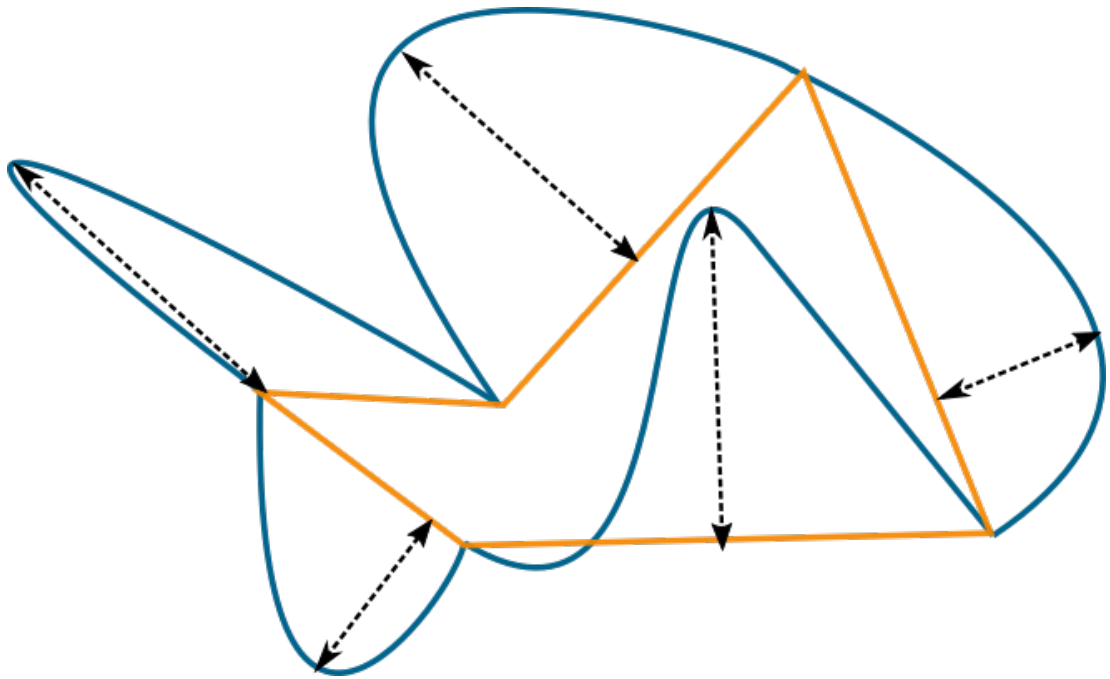
$$d^H(K_1, K_2) = \max\Big(\sup_{x \in K_1} d(x, K_2), \sup_{x \in K_2} d(x, K_1) \Big)$$

Control the surface approximation

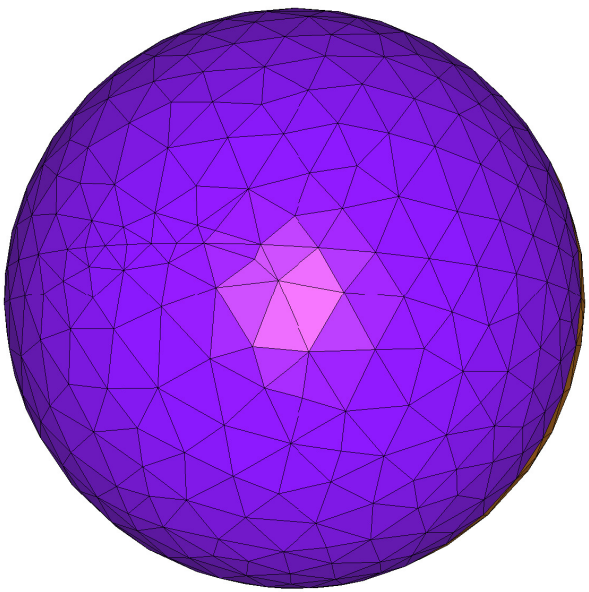
Let T be a conformal mesh in $\mathbb{R}^d$ of the continuous domain Ω.
Let S_T be the surface mesh and $\partial\Omega$ the boundary of Ω

Control the surface approximation

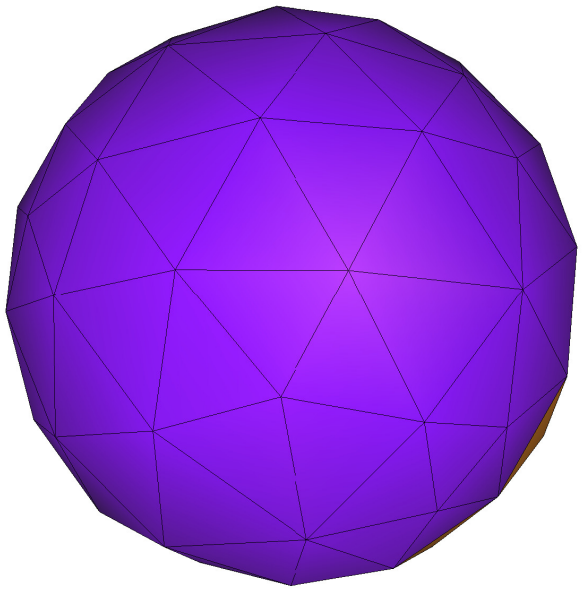
$$d^H(\partial\Omega, S_T) \leq \epsilon$$



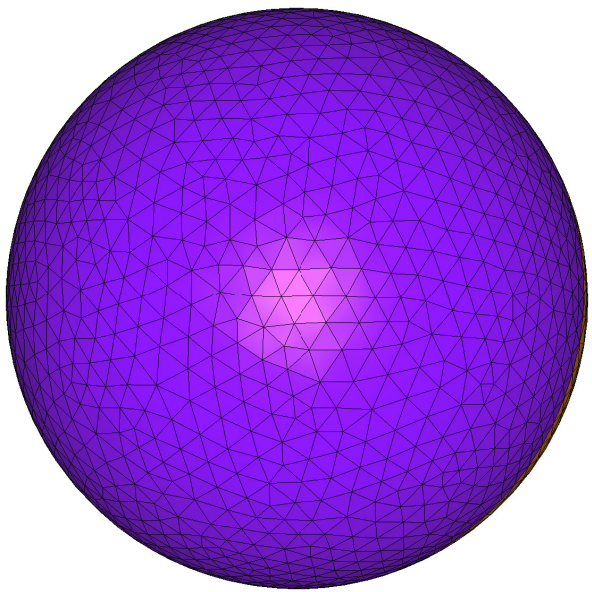
Control the surface approximation : example



init mesh



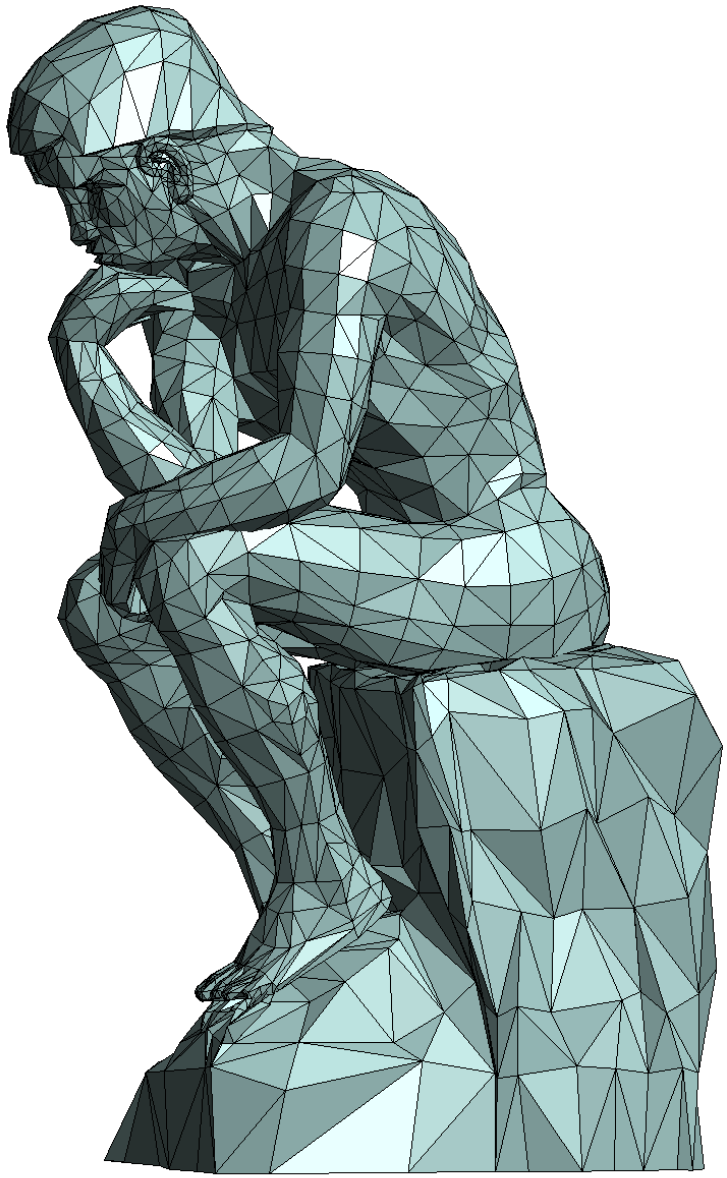
3



0.1

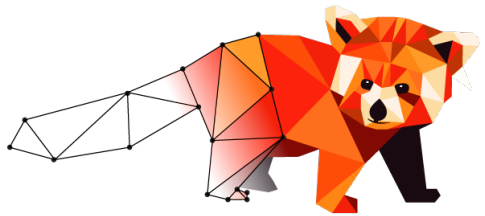
Approximation of a sphere of radius 40 depending on the Hausdorff distance tolerance

Surface mesh improvement



upgrade your meshes

Mesh improvement/adaptation



upgrade
your meshes

Overview

Goals

1. Mesh modifications in agreement to a size map
2. Generation of equilateral elements

Overview

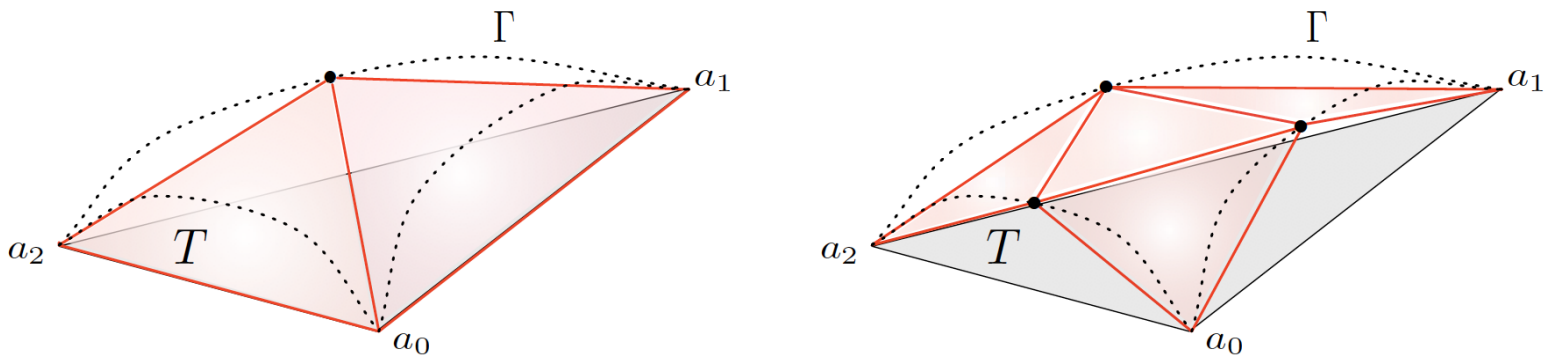
Algorithm

1. Surface analysis
2. Rough surface mesh modifications for a good 'sampling' of the surface :
   - split too long edges (pattern)
   - collapse too short edges
   - swap too bad elements
3. Size map construction :
   - Surface and volume size maps intersection
   - Size map gradation : $\frac{1}{hgrad} < \frac{l_1}{l_2} < hgrad$
4. Rough surface mesh modifications w.r. to the size function
5. Fine mesh modifications w.r. to the size function :
   - split (pattern for surface, Delaunay for volume) or collapse edges
   - swap too bad elements
   - vertex relocation for quality improvement

Surface mesh insertion based on patterns

Algorithm

- identification of all the edges to split
- node insertion on the ideal surface Γ



Splitting of one (left) or three edges (right) of triangle T

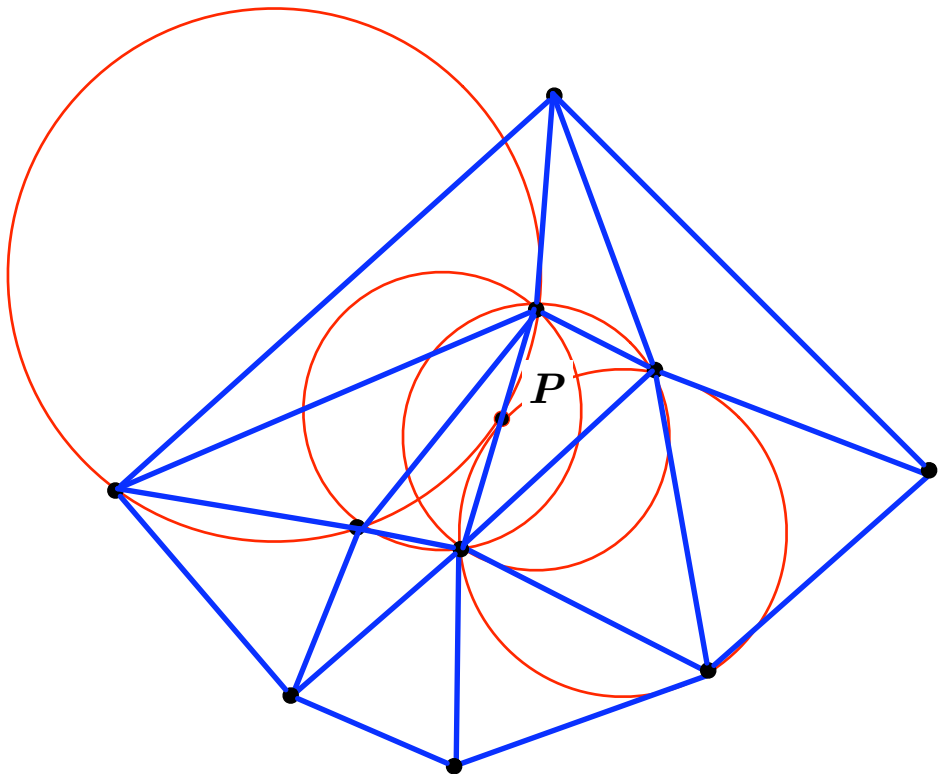
Volume mesh insertion

Delaunay measurement

$$\alpha(T, P) = \frac{d(P, O_T)}{r_T}$$

Cavity definition

$$K \in \mathcal{C}_P \text{ ssi } \alpha(T, P) \leq 1.$$



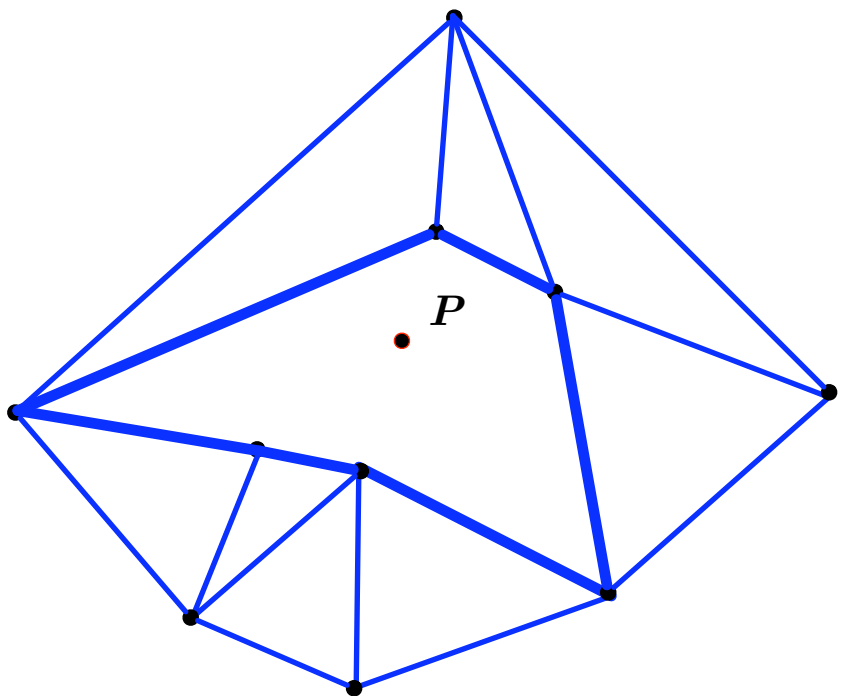
Mesh insertion

Delaunay measurement

$$\alpha(T, P) = \frac{d(P, O_T)}{r_T}$$

Cavity definition

$$T \in \mathcal{C}_P \text{ ssi } \alpha(T, P) \leq 1.$$



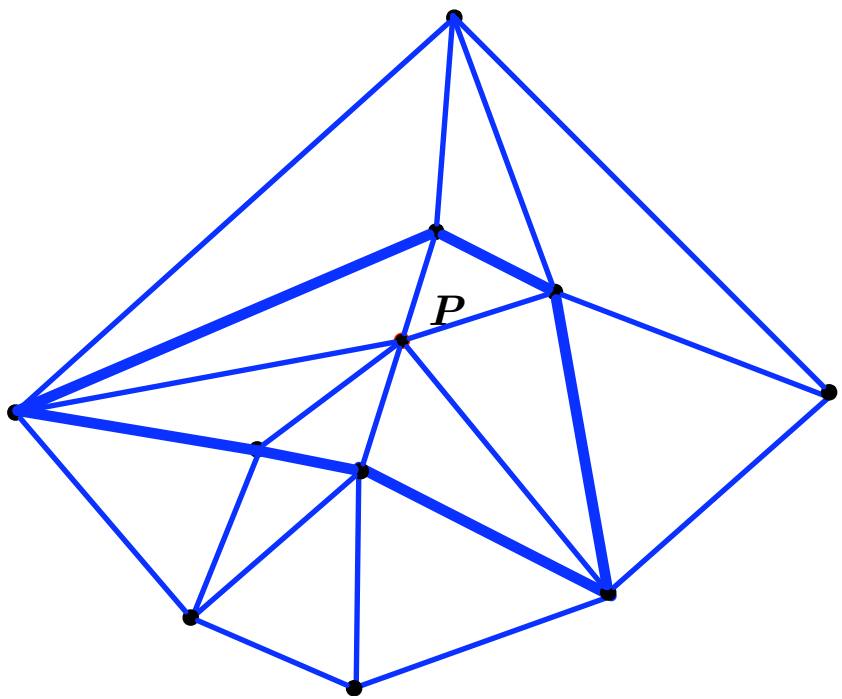
Mesh insertion

Delaunay measurement

$$\alpha(T, P) = \frac{d(P, O_T)}{r_T}$$

Cavity definition

$$T \in \mathcal{C}_P \text{ ssi } \alpha(T, P) \leq 1.$$



Mesh insertion : Anisotropic extension

Delaunay measurement

$$\alpha(T,P) = \frac{d(P,O_T)}{r_T}$$

Cavity definition

$$T \in \mathcal{C}_P \text{ ssi } \alpha(T,P) \leq 1.$$

Anisotropic extension

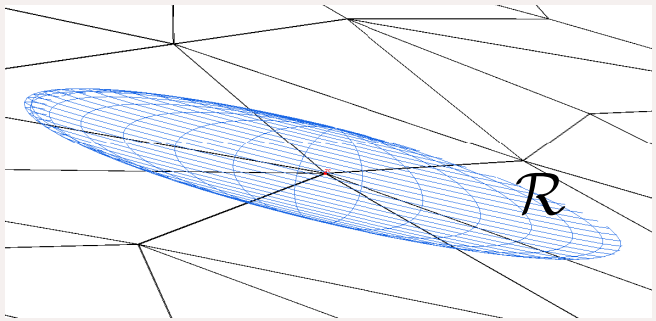
$$\alpha(T,P)_{\mathcal{M}} = \frac{\ell_{\mathcal{M}}(P,O_T)}{r_T}\,.$$

Anisotropic edge length

Metric tensor

$\mathcal{M}$ symetric definite positive tensor :

$$\mathcal{M} = \mathcal{R} \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \mathcal{R}^{-1}$$



eigenvectors related to the edge directions
and λ_i related to the edge lenghts.

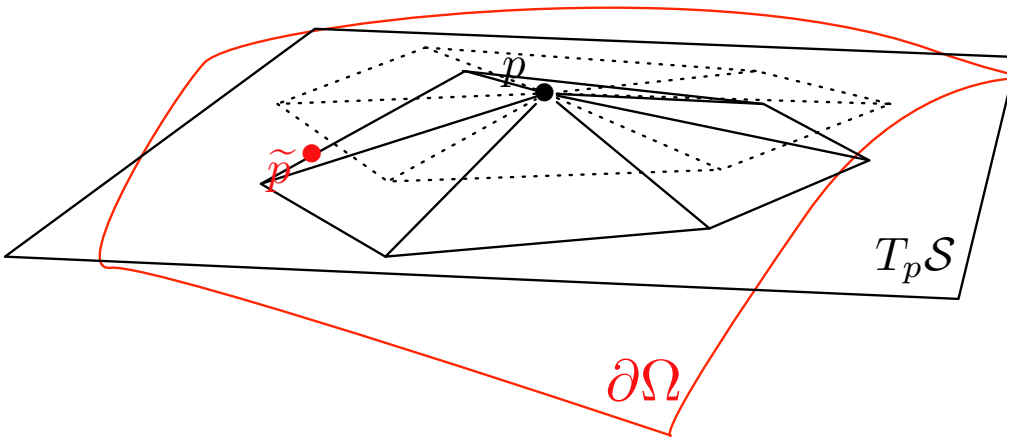
Length edge $\overrightarrow{e}$

$$\ell_{\mathcal{M}}(\overrightarrow{e}) = \|\overrightarrow{e}\|_{\mathcal{M}} = \sqrt{\langle \overrightarrow{e}, \mathcal{M}\overrightarrow{e} \rangle}$$

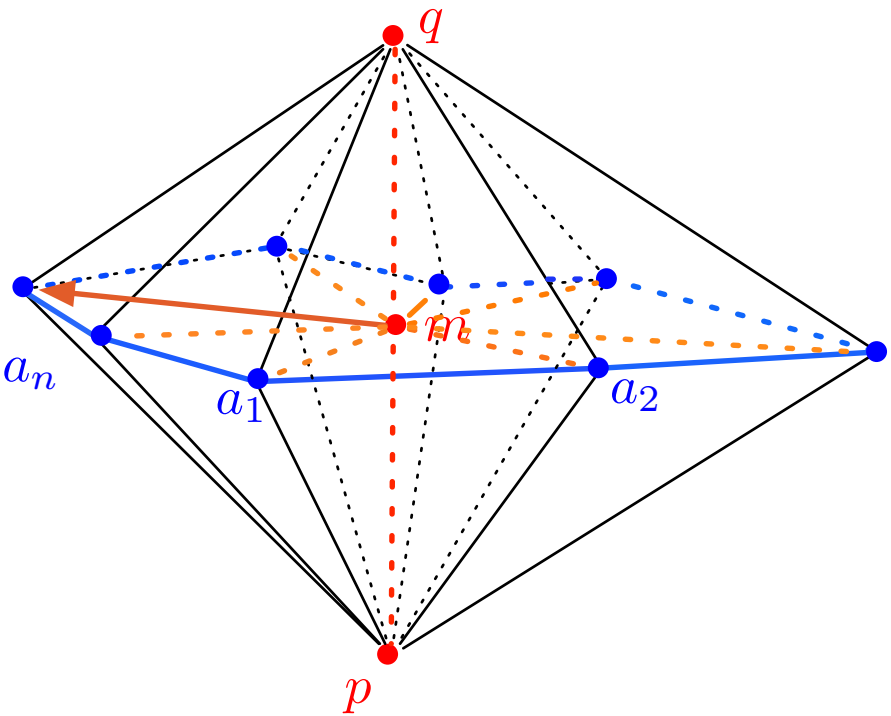
Anisotropic mesh adaptation

Use the length edge definition to decide how to apply the local operators.

Point relocation/Swaps



Surface point relocation



Generic swap

Mesh adaptation for immersed boundary method

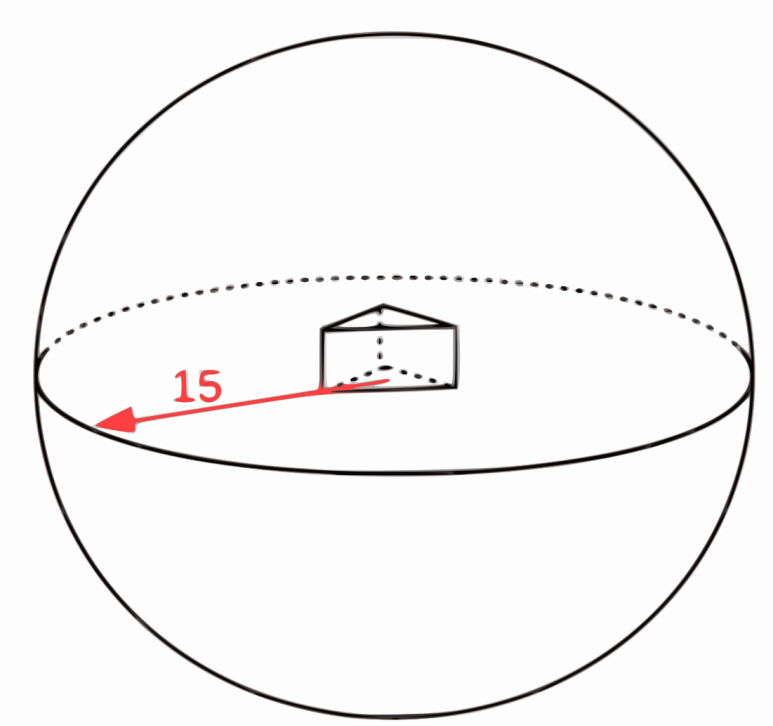
L. Nouveau, Cardamom team

Compressible Navier-Stokes simulation

Mach number 2. ; Re 50 0000

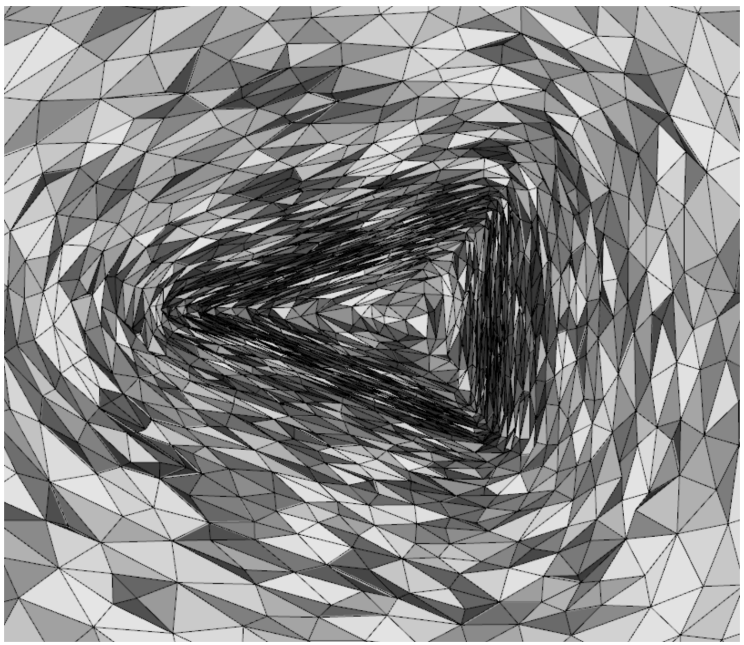
Shock and drag capture

Accurate solution with a minimal number of nodes and elements

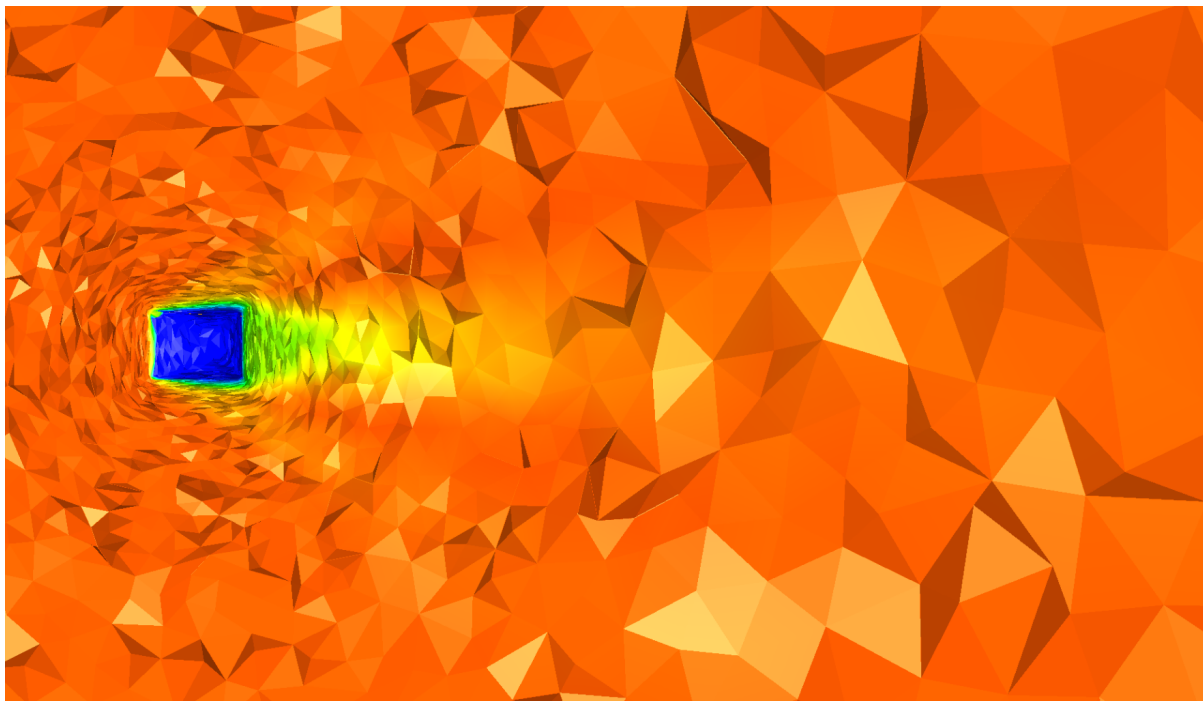


Mesh adaptation for immersed boundary method

L. Nouveau, Cardamom team



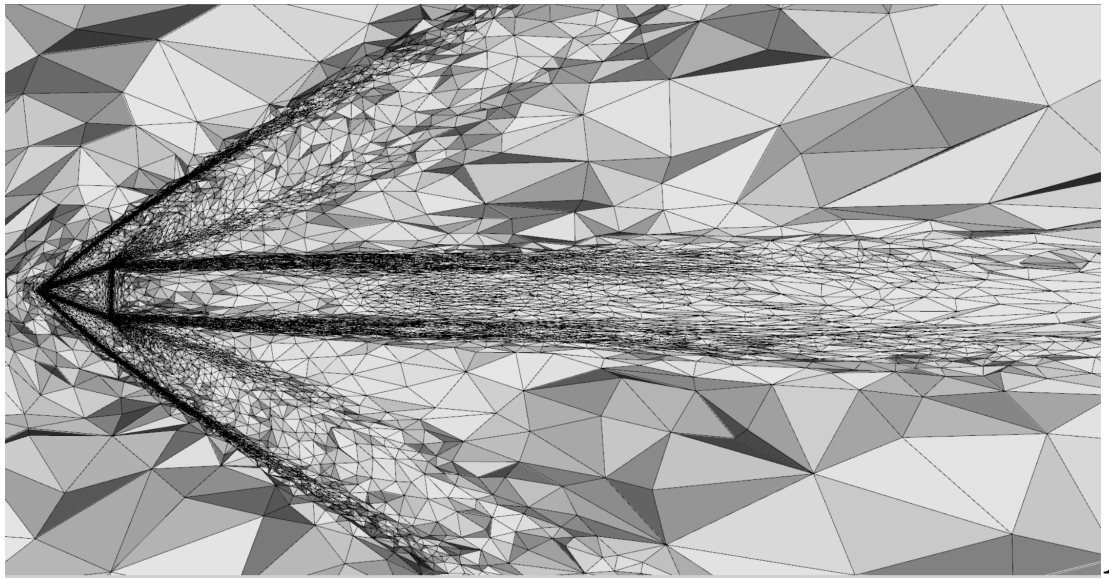
Initial mesh



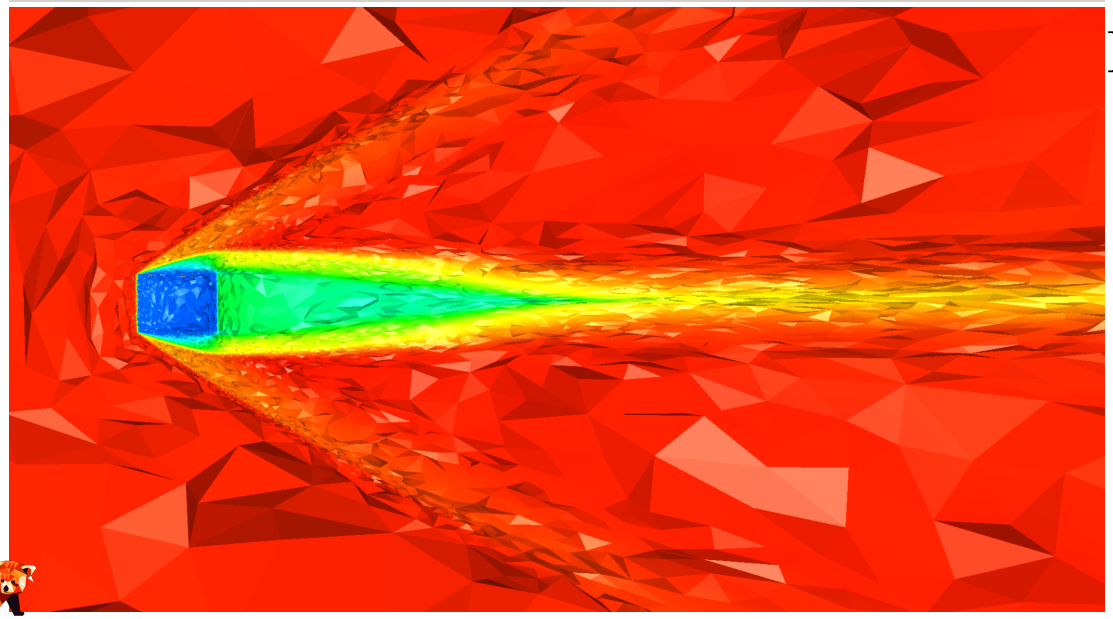
Initial velocity solution

Mesh adaptation for immersed boundary method

L. Nouveau, Cardamom team

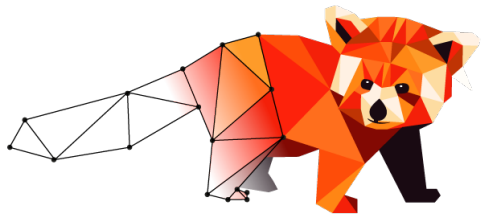


Adapt mesh



Final velocity solution

Level-set discretization



Upgrade
your meshes

Overview

Goals

Obtain an explicit mesh of a specific value of an implicit fonction

Why ?

1. For structural optimization using level-set method
2. To generate a mesh from a bathymetry
3. To limit numerical errors over a front...

Definition : signed distance function

Be Ω, a subset of a metric space with metric d, the signed distance function is the function f such as :

$$f(x) = \begin{cases} -d(x, \partial\Omega) & \text{if } x \in \Omega \\ +d(x, \partial\Omega) & \text{otherwise} \end{cases}$$

Definition : level-set

Be $f : \mathbb{R}^n \mapsto \mathbb{R}$, a level-set of f is a set $L_c(f)$ such as :

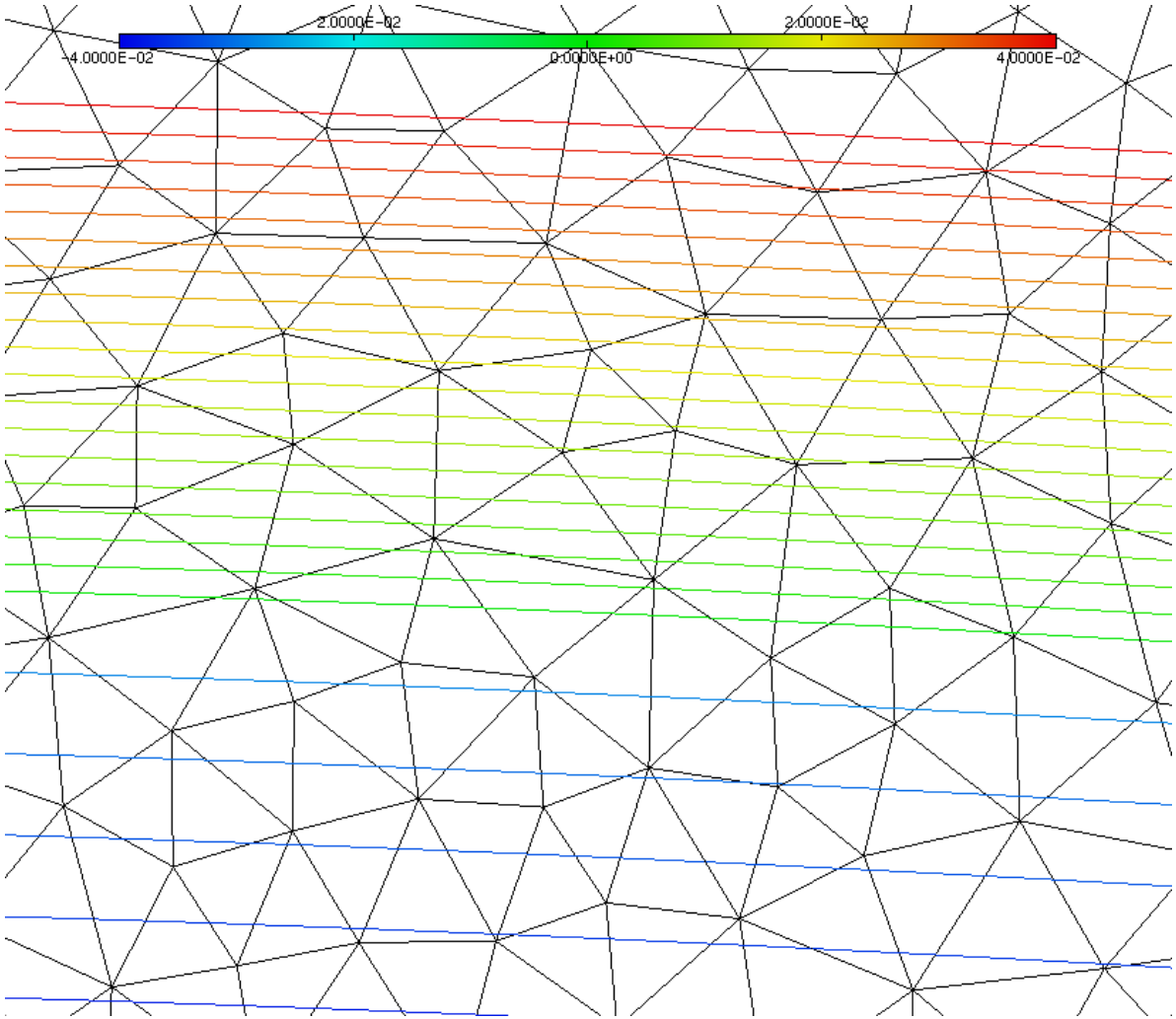
$$L_c(f) = \{(x_1, x_2, ..., x_n) | f(x_1, x_2, ..., x_n) = c\}$$

with c a given constant value.

Implicit function discretization

Input data

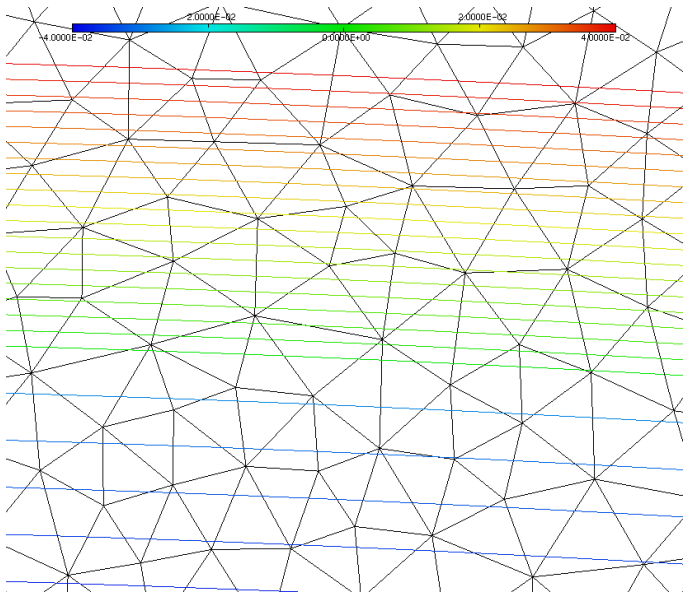
Nodal values of the signed distance function



Implicit function discretization

Steps

1. Mark elements intersected by the level-set
2. For each marked element :
   - 2.1 Mark the edges intersected by the level-set
   - 2.2 Insert a new point at the intersection between the level-set and the edge(s)
   - 2.3 Split the element using patterns and tag the boundary edge
3. Improve the mesh

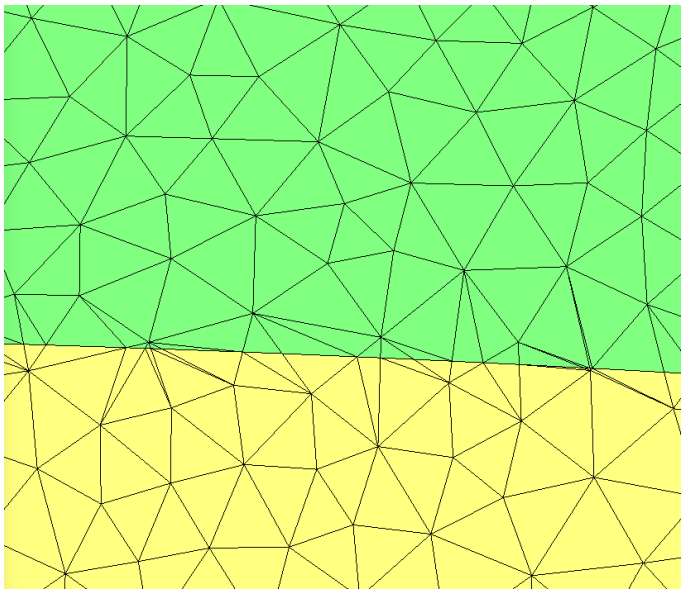


upgrade your meshes

Implicit function discretization

Steps

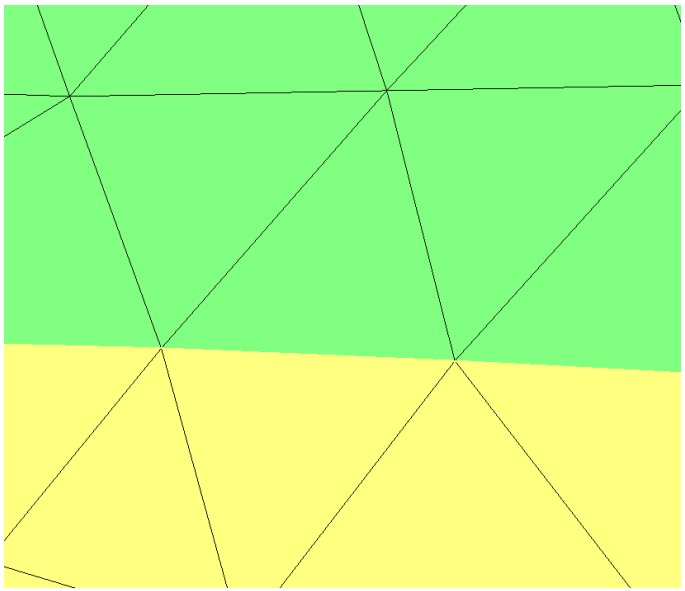
1. Mark elements intersected by the level-set
2. For each marked element :
   - 2.1 Mark the edges intersected by the level-set
   - 2.2 Insert a new point at the intersection between the level-set and the edge(s)
   - 2.3 Split the element using patterns and tag the boundary edge
3. Improve the mesh



Implicit function discretization

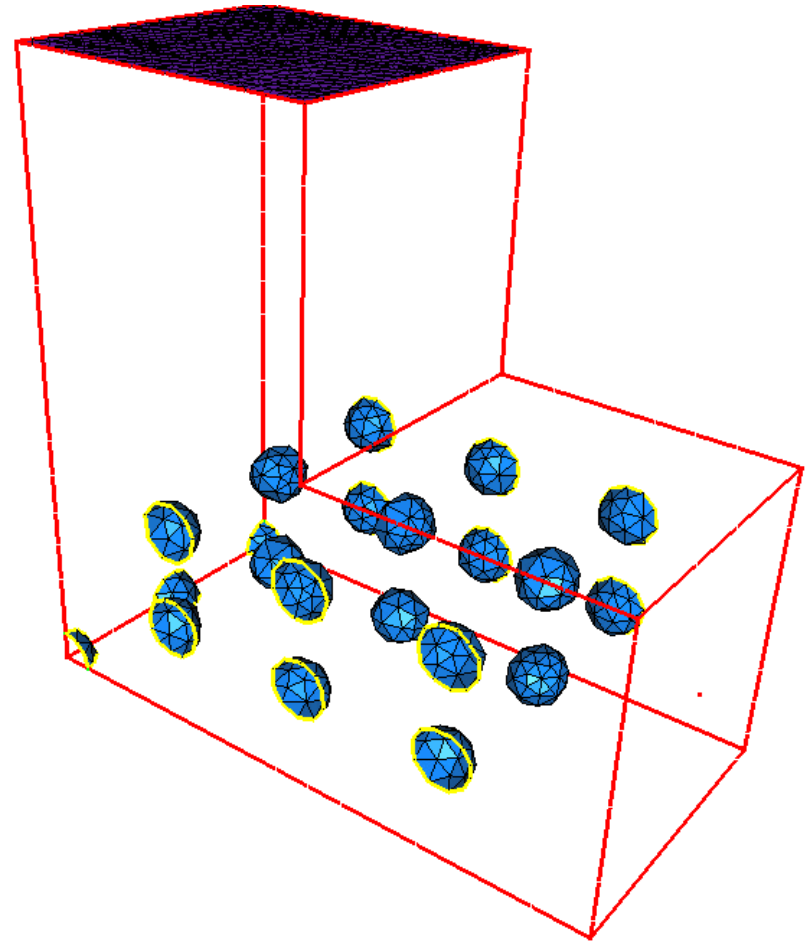
Steps

1. Mark elements intersected by the level-set
2. For each marked element :
   - 2.1 Mark the edges intersected by the level-set
   - 2.2 Insert a new point at the intersection between the level-set and the edge(s)
   - 2.3 Split the element using patterns and tag the boundary edge
3. Improve the mesh



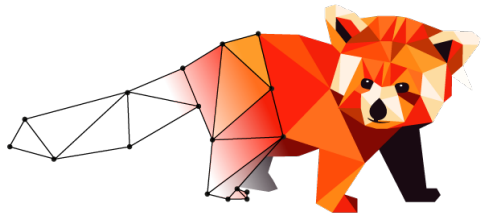
Structural optimization

C. Dapogny, LJK, CNRS



Shape optimization of a L-Beam (stress-based criterion)

Mmg platform in action



upgrade
your meshes

How to use Mmg ?

With command line interface

```
$ mmg3d_O3 dirtyMesh.mesh -o wonderfulMesh.mesh
```

Inside your code,

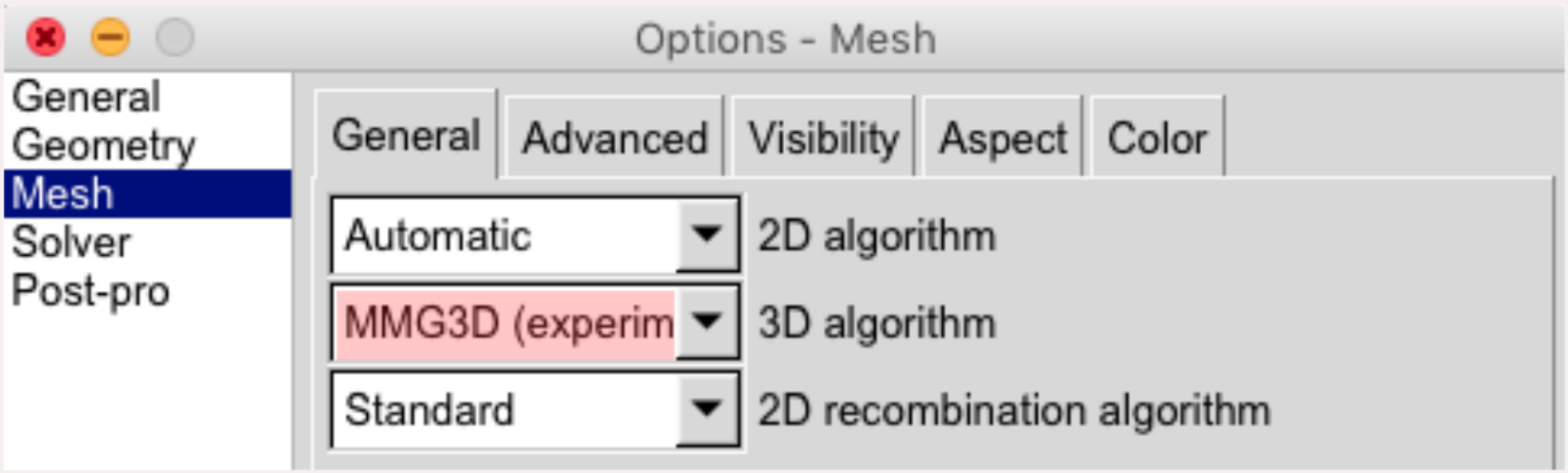
calling our API functions and linking Mmg as external library

```
MMG3D_Init_mesh(MMG5_ARG_start,
                MMG5_ARG_ppMesh,&mmgMesh,MMG5_ARG_ppMet,&mmgSol,
                MMG5_ARG_end);

MMG3D_loadMesh(mmgMesh,"dirtyMesh.mesh");
MMG3D_mmg3dlib(mmgMesh,mmgSol);
MMG3D_saveMesh(mmgMesh,"wonderfulMesh.mesh");

MMG3D_Free_all(MMG5_ARG_start,
               MMG5_ARG_ppMesh,&mmgMesh,MMG5_ARG_ppMet,&mmgSol,
               MMG5_ARG_end);
```

How to use Mmg ?

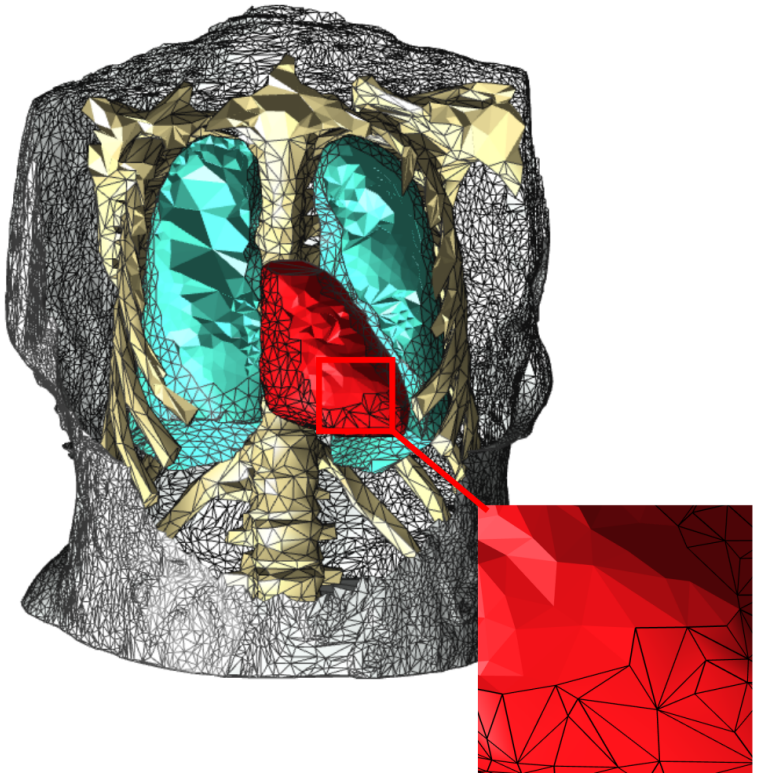
Through other softwares embedding our libraries

- FreeFem++
- Yales2
- AVBP
- Gmsh



Mesh improvement for ECG simulations

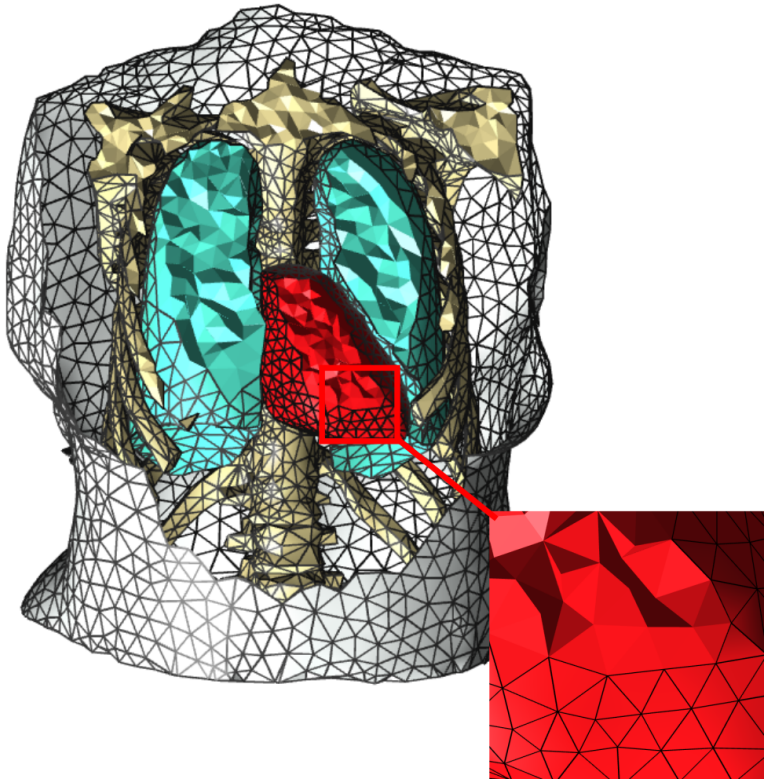
N. Zemzemi, Carmen team, Inria



Input mesh

CT-scan segmentation

278 543 Tetrahedra



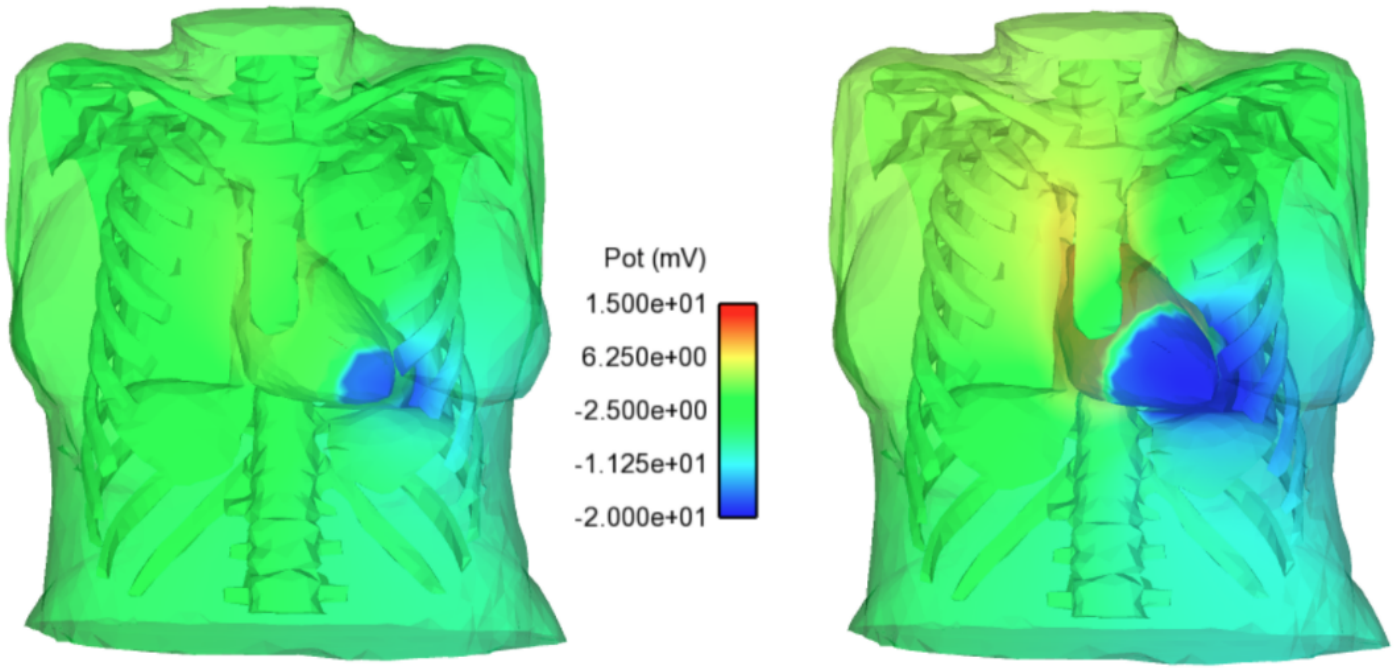
Computational mesh

produced by Mmg3d

128 661 Tetrahedra

Mesh improvement for ECG simulations

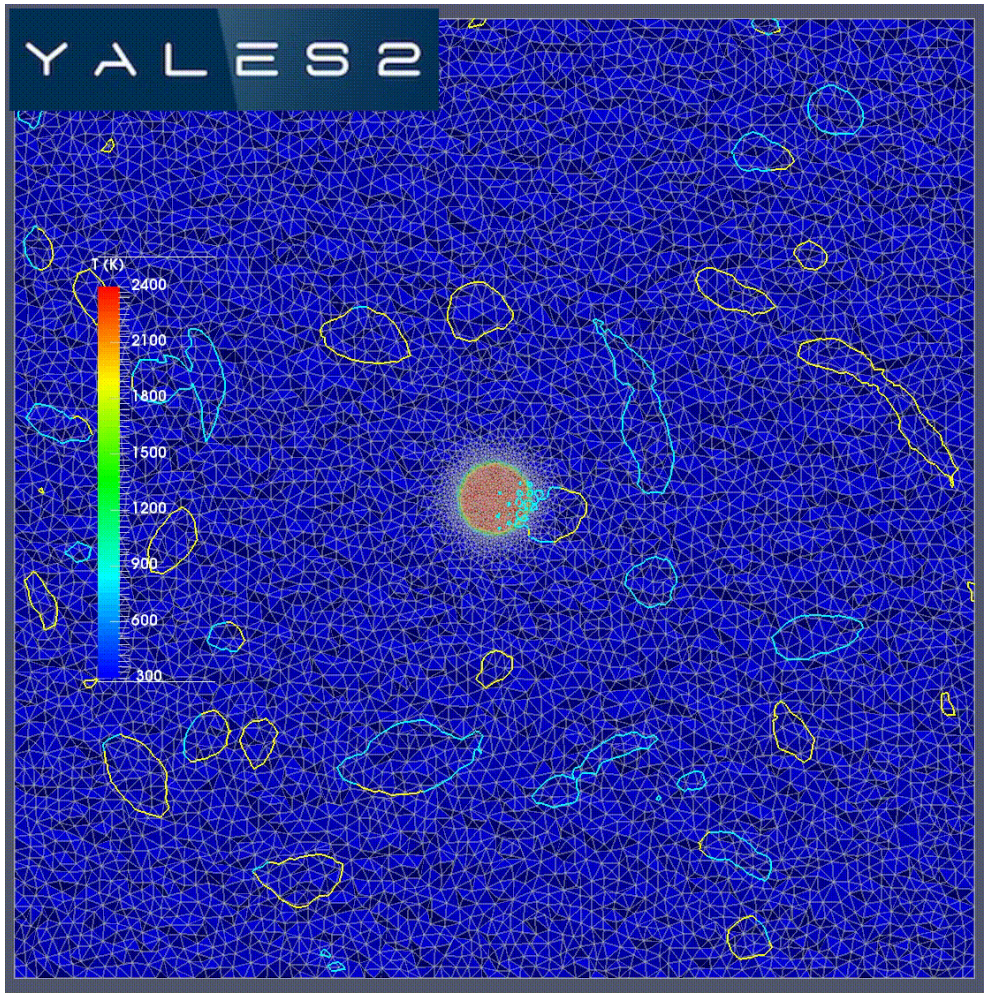
N. Zemzemi, Carmen team, Inria



Solution of the electrocardiography imaging (ECGI)
Potential distribution on the heart surface and the torso volume
at times 16 ms (left) and 40 ms (right)

Isotropic mesh adaptation for LES methods

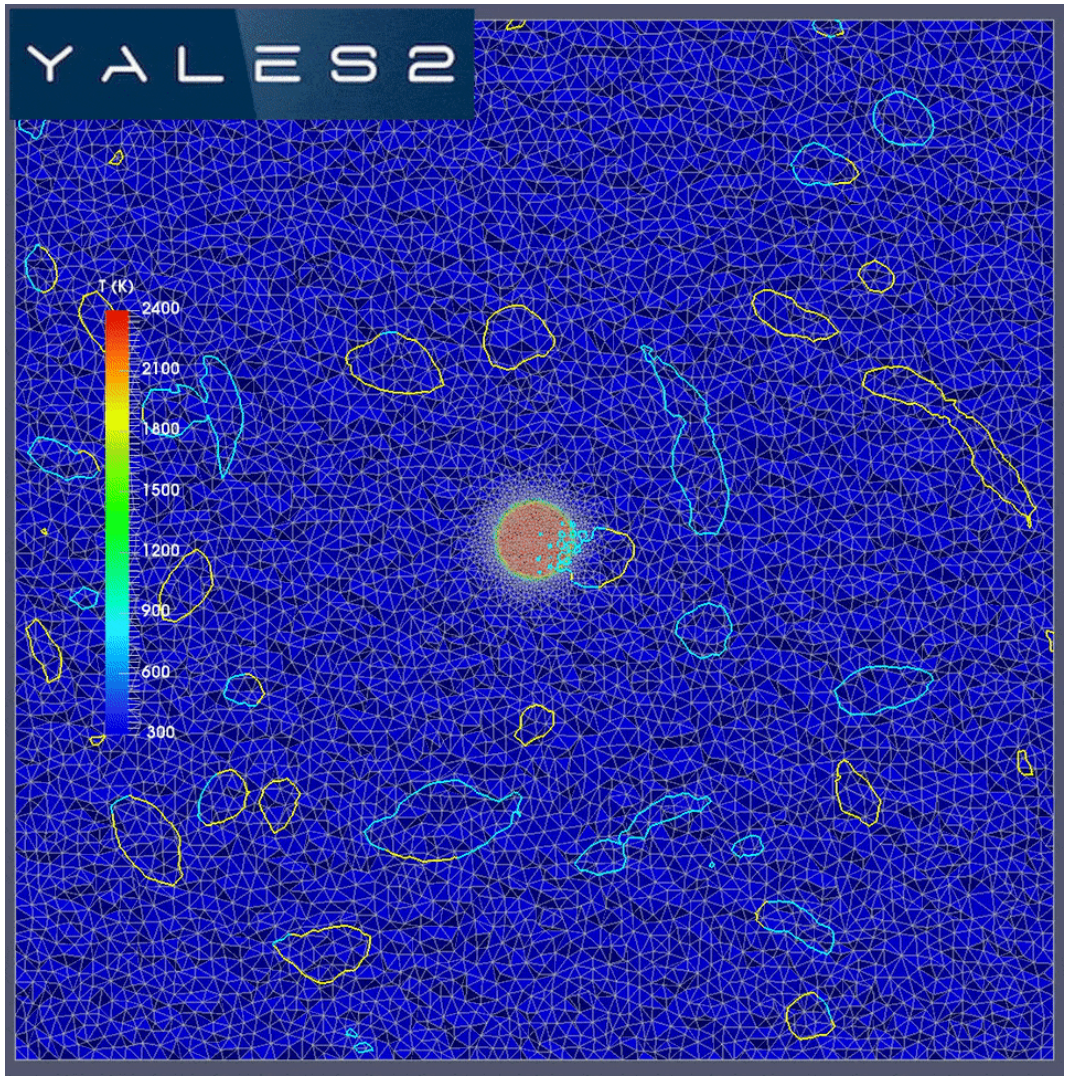
V. Moureau & G. Lartigues, Coria



Turbulent flame kernel propagation without adaptation

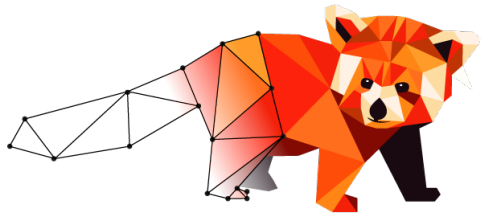
Isotropic mesh adaptation for LES methods

V. Moureau & G. Lartigues, Coria



Turbulent flame kernel propagation with adaptation

open source community



upgrade
your meshes

OS community

Website : http ://www.mmgtools.org

news, tutorials, provisional roadmap...

The forum : https ://forum.mmgtools.org

ask all your questions