The level set method, mesh evolution and shape optimization

G. Allaire[1], Ch. Dapogny[1,2,3], and P. Frey[2]

[1] CMAP, UMR 7641 École Polytechnique, Palaiseau, France
[2] Laboratoire J.L. Lions, UPMC, Paris, France
[3] Technocentre Renault, Guyancourt

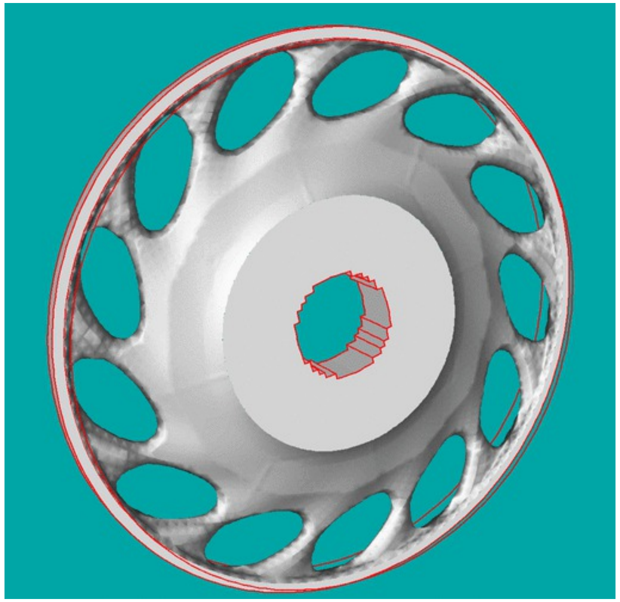
Shape optimization and industrial applications

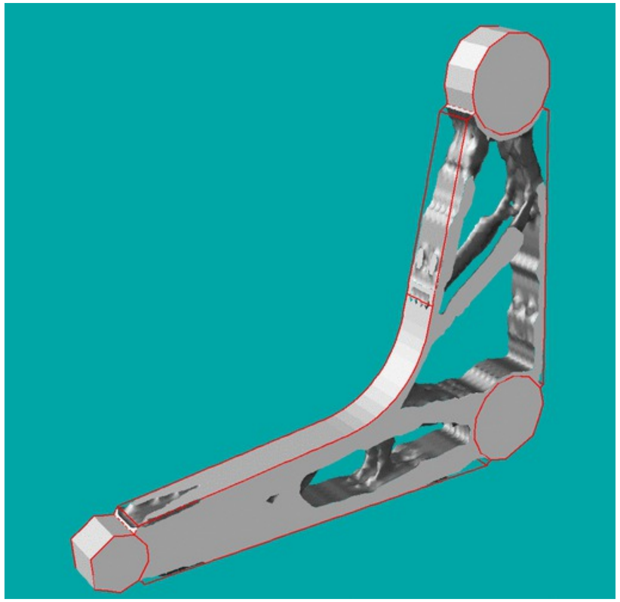
In industry, there is a growing need for optimizing mechanical parts from the early stages of design.

Such problems are difficult, because

- they are highly dependent on the mechanical(s) problem(s) at stake.
- they require an accurate description of the various shapes that could be obtained through the optimization process.

Basically, engineers work by trial and error, and highly rely on physical intuition, but automatic techniques that could lead to non-intuitive designs would prove much more efficient.





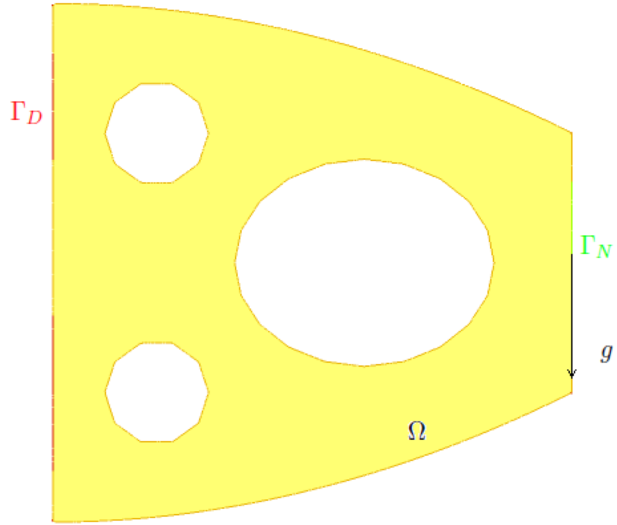
A model problem in linear elasticity

A structure (or shape) is represented by a bounded open domain $\Omega \subset \mathbb{R}^d$, fixed on a part $\Gamma_D \subset \partial\Omega$ of its boundary, and submitted to traction loads g (for the sake of simplicity, body forces are omitted), to be applied on $\Gamma_N \subset \partial\Omega$, $\Gamma_D \cap \Gamma_N = \emptyset$.

The displacement vector field $u_\Omega : \Omega \to \mathbb{R}^d$ is governed by the linear elasticity system:

$$\left\{ \begin{array}{rcll} -div(Ae(u_\Omega)) & = & 0 & \text{in } \Omega \\ u_\Omega & = & 0 & \text{on } \Gamma_D \\ Ae(u_\Omega).n & = & g & \text{on } \Gamma_N \\ Ae(u_\Omega).n & = & 0 & \text{on } \Gamma := \partial\Omega \setminus (\Gamma_D \cup \Gamma_N) \end{array} \right. ,$$

where $e(u) = \frac{1}{2}({}^t\nabla u + \nabla u)$ is the strain tensor field, $Ae(u) = 2\mu e(u) + \lambda tr(e(u))I$ is the stress tensor, and λ, μ are the Lamé coefficients of the material.



A 'Cantilever'

The deformed cantilever

A model problem in linear elasticity

goal : Given an initial structure Ω_0, find a new domain Ω that minimizes a certain functional of the domain $J(\Omega)$, under a volume constraint.

Examples :

- The work of the external loads g or compliance $C(\Omega)$ of domain Ω :

$$C(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega) dx = \int_{\Gamma_N} g.u_\Omega \, ds$$

- A criterion $S(\Omega)$ based on the norm of the stress tensor $\sigma(u_\Omega)$ of the shape:

$$S(\Omega) = \int_\Omega k(x) ||\sigma(u_\Omega)||^2 dx,$$

  where $k(x)$ is a weight factor.

The volume constraint is enforced with a fixed Lagrange multiplier ℓ :

$\Rightarrow$ minimize $J(\Omega) := C(\Omega) + \ell \, Vol(\Omega)$, or $S(\Omega) + \ell \, Vol(\Omega)$.

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain : Hadamard's method
2. The 'classical' level set method, and the proposed strategy

II. From meshed domains to a level set description,... and conversely

1. Initializing level set functions with the signed distance function
2. Meshing the negative subdomain of a level set function : local remeshing

III. Application to shape optimization

1. Numerical Implementation
2. The algorithm in motion
3. Some numerical results

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain : Hadamard's method
2. The 'classical' level set method, and the proposed strategy

II. From meshed domains to a level set description,... and conversely

1. Initializing level-set functions with the signed distance function
2. Meshing the negative subdomain of a level set function : local remeshing

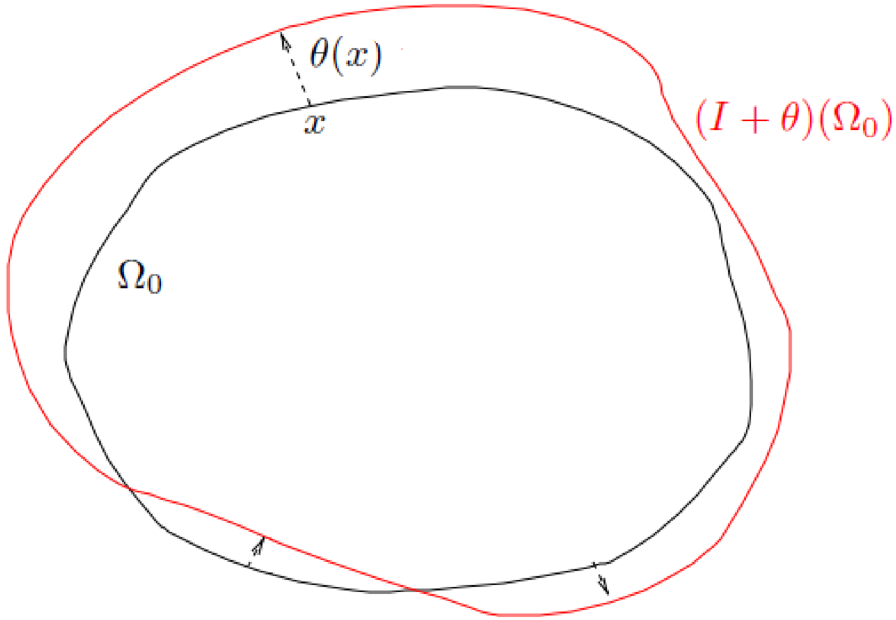
III. Application to shape optimization

1. Numerical Implementation
2. The algorithm in motion
3. Some numerical results

Differentiation with respect to the domain : Hadamard's method

Given a reference (smooth enough) domain Ω_0, we consider variations of Ω_0 of the form:

$$(I+\theta)(\Omega_0), \;\; \theta \in W^{1,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right).$$



DEFINITION 1 *A functional $\Omega \mapsto J(\Omega) \in \mathbb{R}$ is shape differentiable at Ω_0 if*

$$W^{1,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right) \ni \theta \mapsto J((I+\theta)(\Omega_0))$$

is Fréchet-differentiable at 0, i.e. we have the following expansion, in the vicinity of 0:

$$J((I+\theta)(\Omega_0)) = J(\Omega_0) + J\prime(\Omega_0)(\theta) + o(||\theta||)$$

Differentiation with respect to the domain : Hadamard's method

- The Structure theorem gives the form of the shape derivative of a rather general class of functionals $J(\Omega)$.

$$J\prime(\Omega)(\theta) = \int_\Gamma v\,(\theta.n)\,ds,$$

where v is a scalar field defined on Γ.

For instance, in the case of the compliance $J(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega)\,dx$,

$$v = -Ae(u_\Omega) : e(u_\Omega).$$

- This shape gradient provides a natural descent direction θ for functional J. Defining $\theta = -vn$ yields, for $t > 0$ sufficiently small (*to be found numerically*):

$$J((I + t\theta)(\Omega)) = J(\Omega) - t\int_\Gamma v^2 ds + o(t) < J(\Omega).$$

The generic numerical algorithm

Gradient algorithm : For $n = 0, \ldots$ convergence,

1. Compute the solution u_{Ω^n} of the above elasticity system of Ω^n.
2. Compute the shape derivative $J'(\Omega^n)$ thanks to the above formula, and infer a descent direction θ^n for the cost functional.
3. Advect the shape Ω^n according to this displacement field, so as to get Ω^{n+1}.

Problem : We need to

- efficiently advect the shape Ω^n at each step
- get a mesh of each shape Ω^n so as to perform the required finite element computations.

Reconciling both constraints is difficult, the bulk of approaches for moving meshes being heuristic, and at some point limited.

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain : Hadamard's method
2. The 'classical' level set method, and the proposed strategy

II. From meshed domains to a level set description,... and conversely

1. Initializing level-set functions with the signed distance function
2. Meshing the negative subdomain of a level set function : local remeshing

III. Application to shape optimization

1. Numerical Implementation
2. The algorithm in motion
3. Some numerical results

A few words about the level set Method

A paradigm : *When you want to describe a surface evolution, represent it with an implicit function.*

Given a bounded domain $\Omega \subset \mathbb{R}^d$, define it with a function ϕ on the whole $\mathbb{R}^d$ such that

$$\phi(x) \quad < 0 \quad \text{if } x \in \Omega \quad ; \quad \phi(x) \quad = 0 \quad \text{if } x \in \partial\Omega \quad ; \quad \phi(x) \quad > 0 \quad \text{if } x \in {}^c\Omega$$

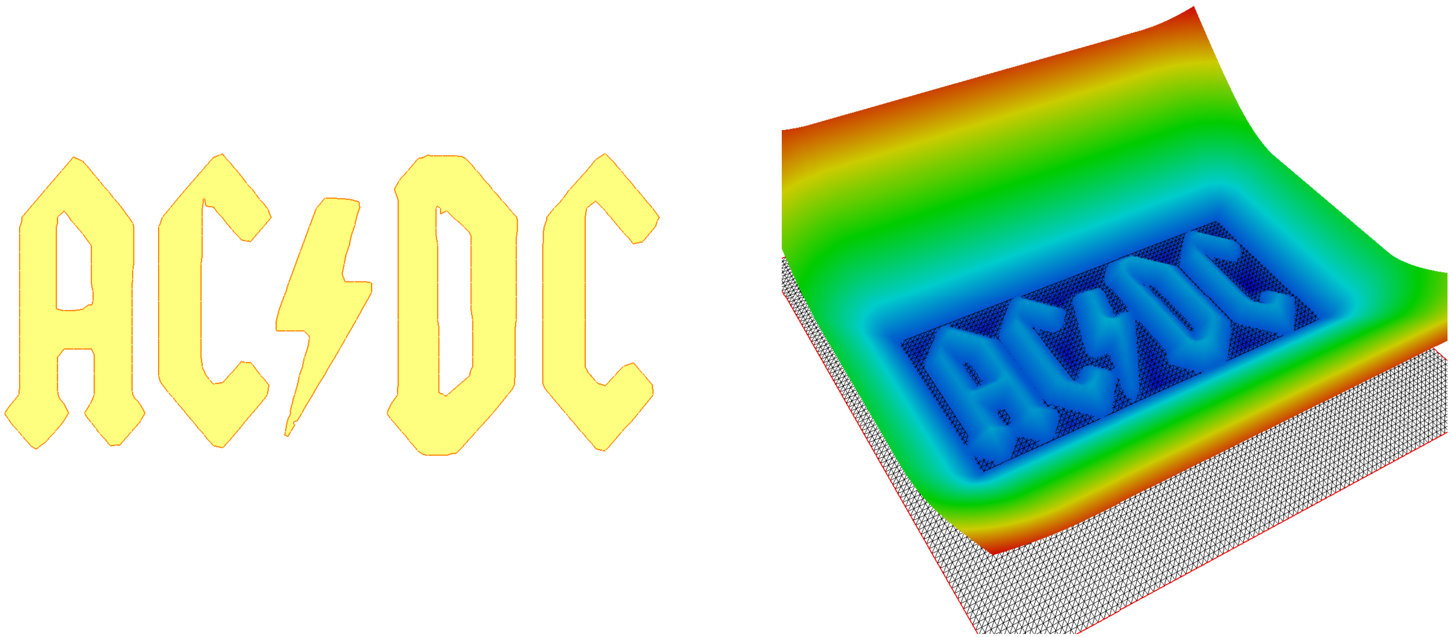


Figure 1: *A bounded domain $\Omega \subset \mathbb{R}^2$ (left), some level sets of an implicit function representing Ω (right).*

Surface evolution equations in the level set framework

Suppose that, for every time t, the domain $\Omega(t) \subset \mathbb{R}^d$ is represented by an implicit function $\phi(t,.)$ on $\mathbb{R}^d$, and is subject to an evolution defined by velocity $v(t,x) \in \mathbb{R}^d$. Then

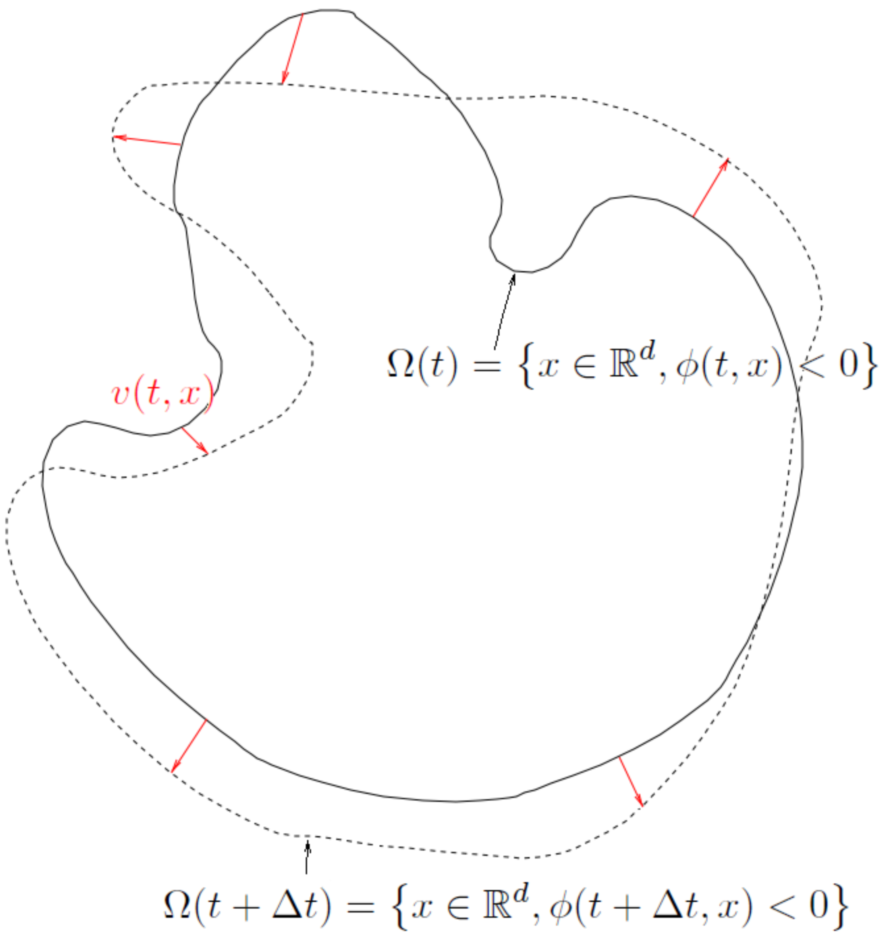
$$\forall t,\ \forall x \in \mathbb{R}^d,\ \frac{\partial \phi}{\partial t}(t,x) + v(t,x).\nabla\phi(t,x) = 0$$

In many applications, the velocity $v(t,x)$ is normal to the boundary $\partial\Omega(t)$:

$$v(t,x) := V(t,x)\frac{\nabla\phi(t,x)}{|\nabla\phi(t,x)|}.$$

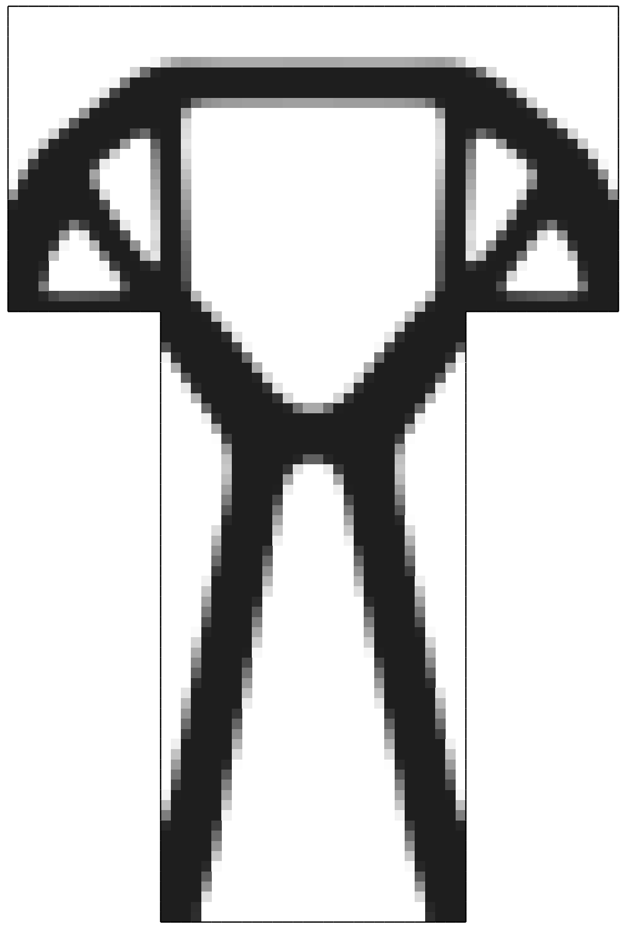
Then the evolution equation rewrites as a Hamilton-Jacobi type equation

$$\forall t,\ \forall x \in \mathbb{R}^d,\ \frac{\partial \phi}{\partial t}(t,x) + V(t,x)|\nabla\phi(t,x)| = 0$$



The level set method of Allaire-Jouve-Toader

- The shapes Ω^n are embedded in a computational box $\mathcal{D}$ equipped with a fixed mesh.
- The successive shapes Ω^n are accounted for in the level set framework, i.e. by the knowledge of a function ϕ^n defined on the whole box $\mathcal{D}$ which implicitly defines them.
- At each step n, the exact linear elasticity system on Ω^n is approximated by the Ersatz material approach : the void $\mathcal{D} \setminus \Omega^n$ is filled with a very 'soft' material, which leads to an approximate linear elasticity system, defined on $\mathcal{D}$.
- This approach is very versatile and does not require an exact mesh of the shapes at each iteration.



Shape accounted for with a level set description

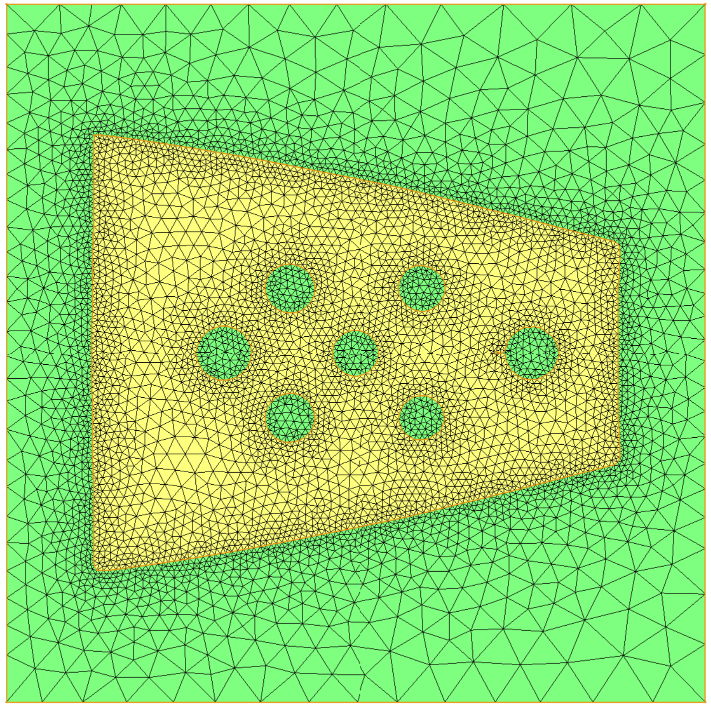
The proposed method

Elaborating on the Level set method of allaire, Jouve and Toader, we propose a slightly different approach which still benefits from the versatility of level set methods to account for large deformations of shapes (even topological changes), but enjoys at each step the knowledge of a mesh of the shape.

The unstructured mesh $\mathcal{T}^n$ of the computational box $\mathcal{D}$ is allowed to evolve so that at each step n, the shape Ω^n is explicitly discretized.

- Level set methods are performed on this unstructured mesh to account for the advection of the shapes $\phi^n \to \phi^{n+1}$.
- Finite element computations are performed on the part on this mesh corresponding to the shape.

$$(\Omega^n, \mathcal{T}^n) \to (\Omega^{n+1}, \mathcal{T}^{n+1}) \quad \Leftrightarrow \quad \phi^n \to \phi^{n+1}$$



Shape equipped with a mesh, conformally embedded in a mesh of the computational box.

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain : Hadamard's method
2. The 'classical' level set method, and the proposed strategy

II. From meshed domains to a level set description,... and conversely

1. Initializing level-set functions with the signed distance function
2. Meshing the negative subdomain of a level set function : local remeshing

III. Application to shape optimization

1. Numerical Implementation
2. The algorithm in motion
3. Some numerical results

Initializing level-set functions with the signed distance function

DEFINITION 2 *Let $\Omega \subset \mathbb{R}^d$ a Lispchitz domain. The signed distance function to Ω is:*

$$d_\Omega(x) = \begin{cases} -d(x, \partial\Omega) & \text{if } x \in \Omega \\ 0 & \text{if } x \in \partial\Omega \\ d(x, \partial\Omega) & \text{if } x \in \overline{^c\Omega} \end{cases}$$

, where $d(\cdot, \partial\Omega)$ is the usual Euclidean distance

- Several algorithms exist to compute the signed distance function to a given domain on an unstructured mesh (e.g. extensions to the Fast Marching, and Fast Sweeping methods).

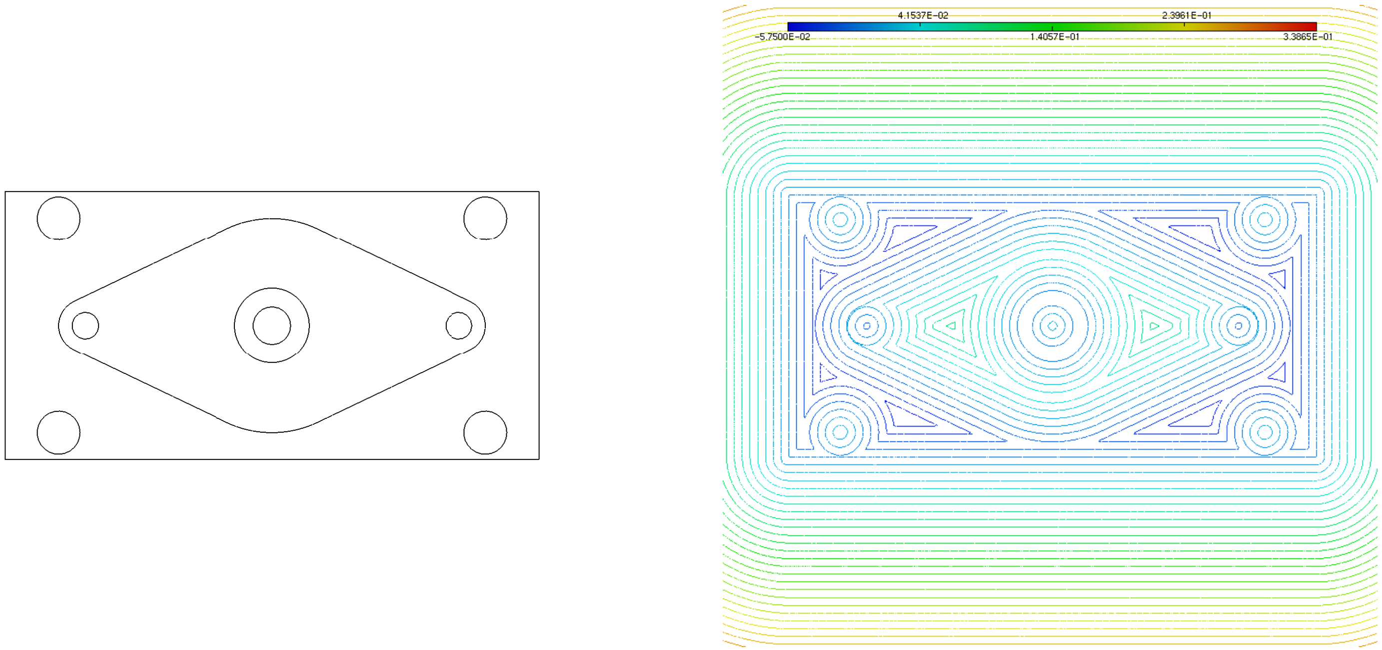


Figure 2: *Computation of the signed distance function to a discrete contour (left), on a fine background mesh.*

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain : Hadamard's method
2. The 'classical' level set method, and the proposed strategy

II. From meshed domains to a level set description,... and conversely

1. Initializing level-set functions with the signed distance function
2. Meshing the negative subdomain of a level set function : local remeshing

III. Application to shape optimization

1. Numerical Implementation
2. The algorithm in motion
3. Some numerical results

Meshing the negative subdomain of a level set function

Discretizing explicitely the 0 level set of a scalar function defined at the vertices of a simplicial mesh $\mathcal{T}$ of a computational box $\mathcal{D}$ is relatively easy, resorting to patterns.

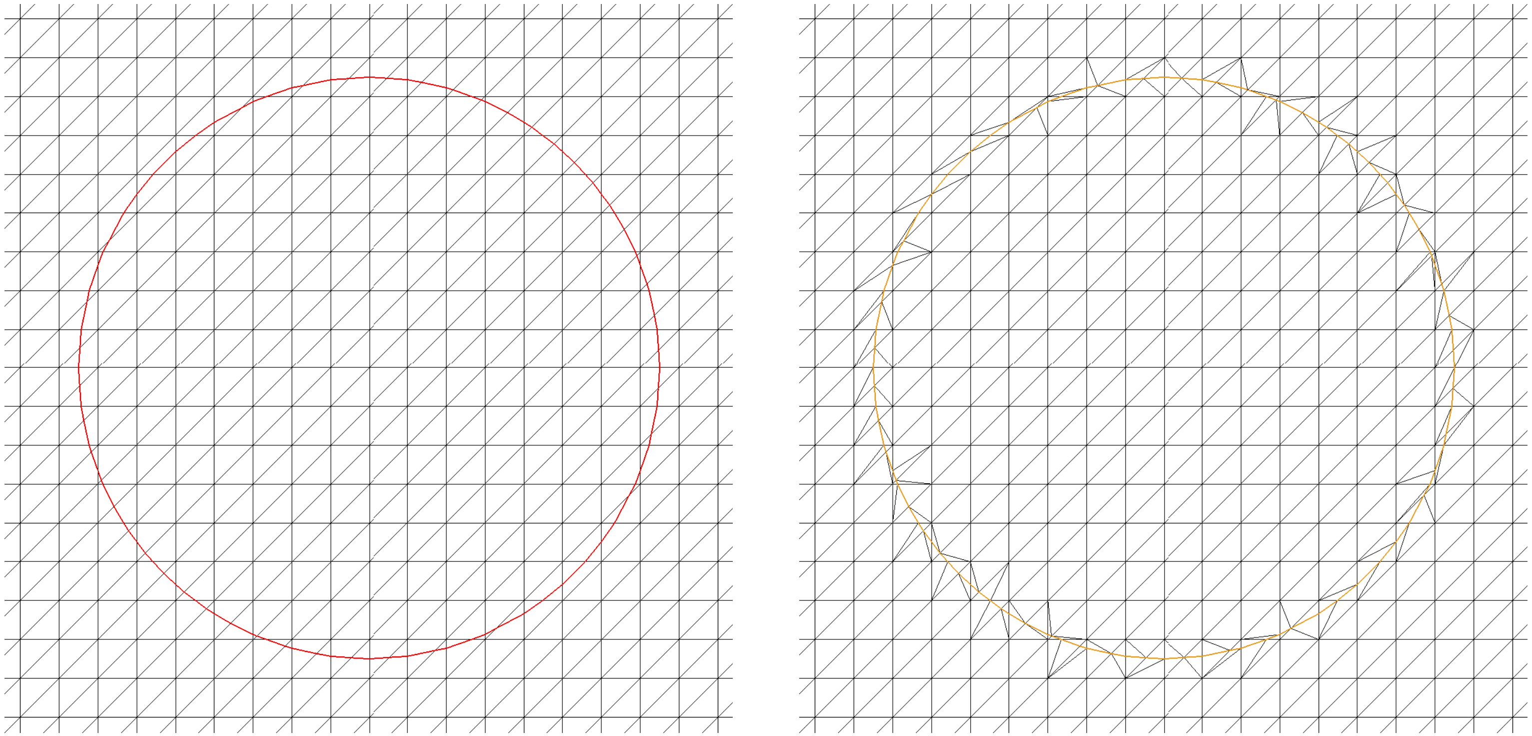


Figure 3: (left) 0 level set of a scalar function defined over a mesh ; (right) explicit discretization in the mesh.

However, doing so is bound to produce a very low-quality mesh, on which finite element computations will prove slow, inaccurate, not to say impossible.

Hence the need to improve the quality of the mesh while retaining its geometric features.

Local remeshing in $3d$

- Let $\mathcal{T}$ an initial - valid, yet potentially ill-shaped - tetrahedral mesh $\mathcal{T}$. $\mathcal{T}$ carries a triangular surface mesh $\mathcal{S}_{\mathcal{T}}$, whose elements appear as faces of tetrahedra of $\mathcal{T}$.
- $\mathcal{T}$ is intended as an approximation of an ideal domain $\Omega \subset \mathbb{R}^3$, and $\mathcal{S}_{\mathcal{T}}$ as an approximation of its boundary $\partial\Omega$.

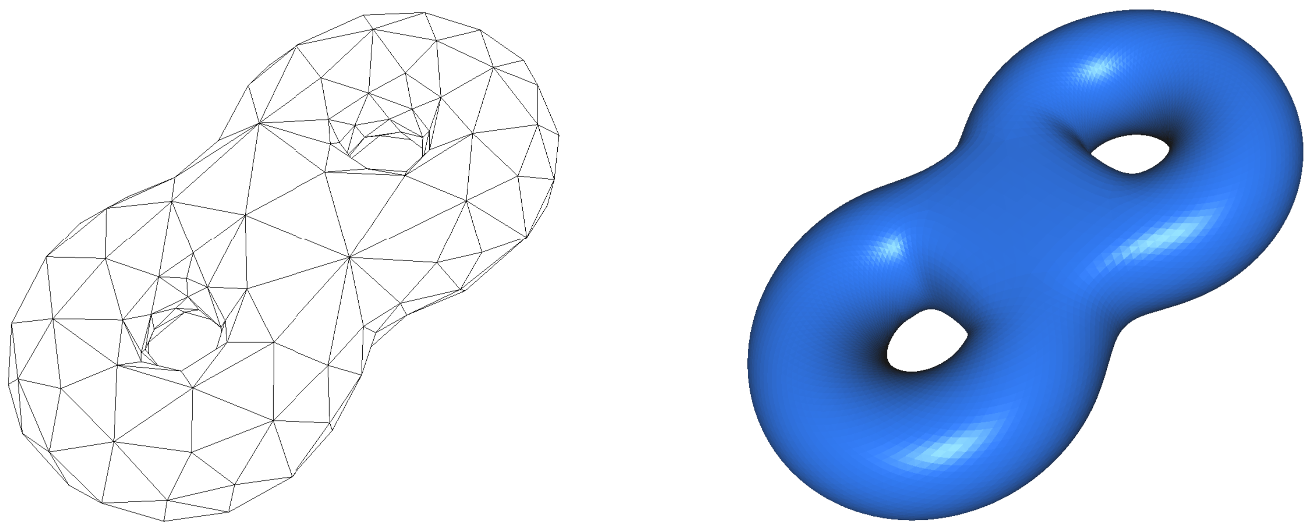


Figure 4: Poor geometric approximation (left) of a domain with smooth boundary (right)

Thanks to local mesh operations, we aim at getting a new, well-shaped mesh $\widetilde{\mathcal{T}}$, whose corresponding surface mesh $\widetilde{\mathcal{S}_{\mathcal{T}}}$ is a good approximation of $\partial\Omega$.

Local remeshing in 3d : definition of an ideal domain

- In realistic cases, the ideal underlying domain Ω associated to $\mathcal{T}$ is unknown.
- However, from the sole data of $\mathcal{T}$ (and $\mathcal{S}_{\mathcal{T}}$), one can reconstruct approximations of geometric features of Ω : sharp angles, normal vectors at regular surface points,...
- These geometric data allow to define rules for the generation of a local parametrization of $\partial\Omega$, around a considered surface triangle $T \in \mathcal{S}_{\mathcal{T}}$, for instance as a Bézier surface.

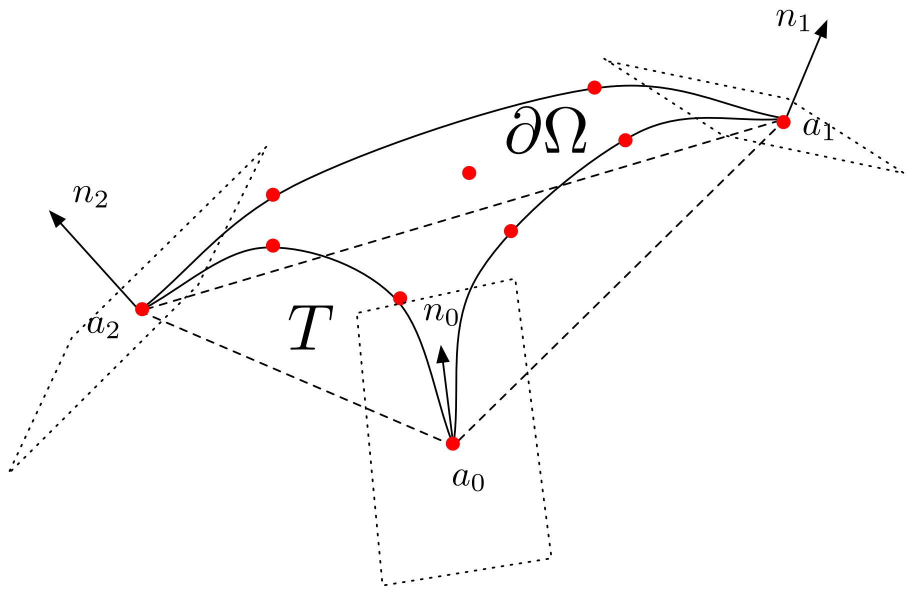


Figure 5: Generation of a cubic Bézier polynomial parametrization for the piece of $\partial\Omega$ associated to triangle T, from the approximated geometrical features (normal vectors at nodes).

Local mesh operators : edge splitting

If an edge pq is too long, insert its midpoint m, then split it into two.

- If pq belongs to a surface triangle $T \in \mathcal{S}_{\mathcal{T}}$, the midpoint m is inserted as the midpoint on the local piece of $\partial\Omega$ computed from T. Else, it is merely inserted as the midpoint of p and q.
- An edge may be 'too long' because it is too long when compared to the prescribed size, or because it causes a bad geometric approximation of $\partial\Omega$,...

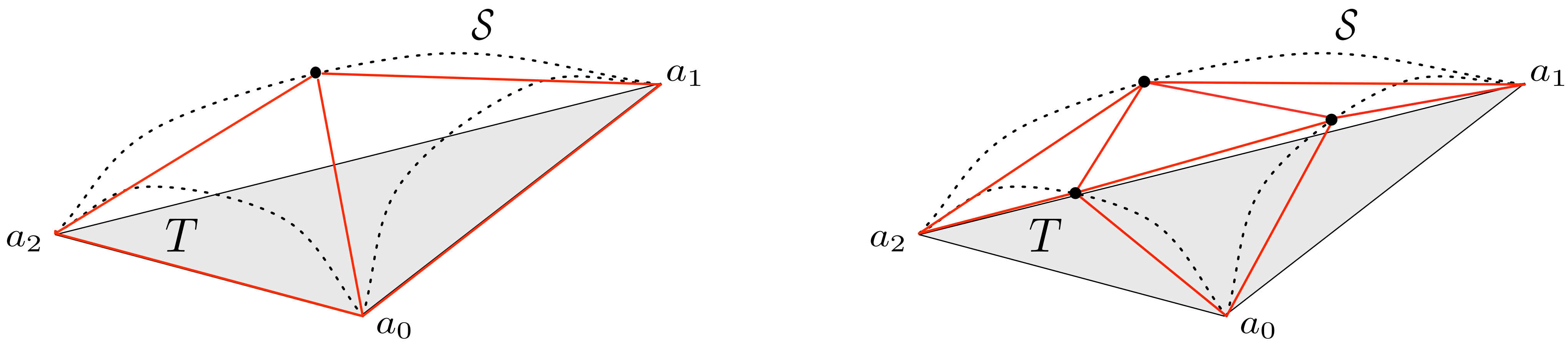


Figure 6: Splitting of one (left) or three (right) edges of surface triangle T, positioning the three new points on the ideal surface $\mathcal{S}$ (dotted).

Local mesh operators : edge collapse

If an edge pq is too short, merge its two endpoints.

- This operation may deteriorate the geometric approximation of $\partial\Omega$, and even invalidate some tetrahedra: some checks have to be performed to ensure the validity of the resulting configuration.
- An edge may be 'too short' because it is too short when compared to the prescribed size, or because it proves unnecessary to a nice geometric approximation of $\partial\Omega$,...

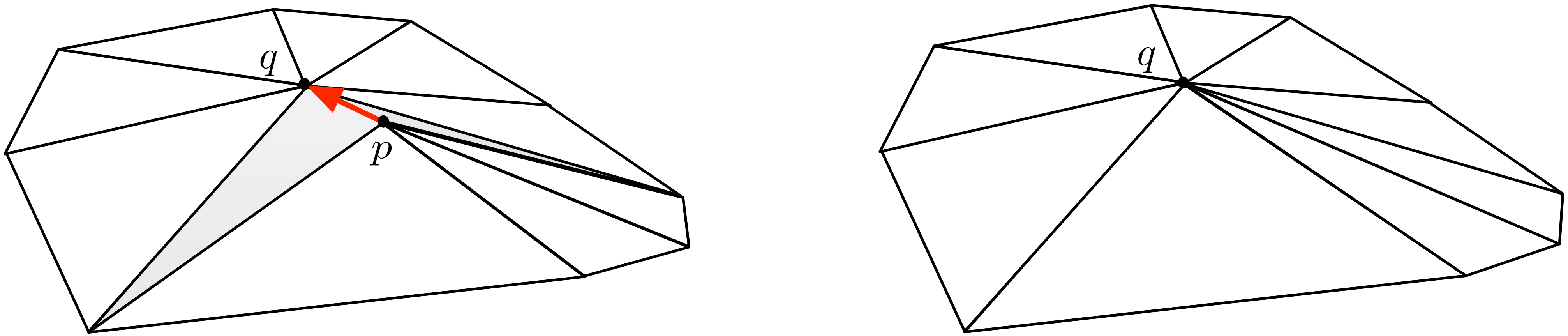


Figure 7: Collapse of point p over q.

Local mesh operators : edge swap, node relocation,...

So as to enhance the global quality of the mesh (or the geometric approximation of $\partial\Omega$), some connectivities can be swapped, and some nodes can be slightly moved.

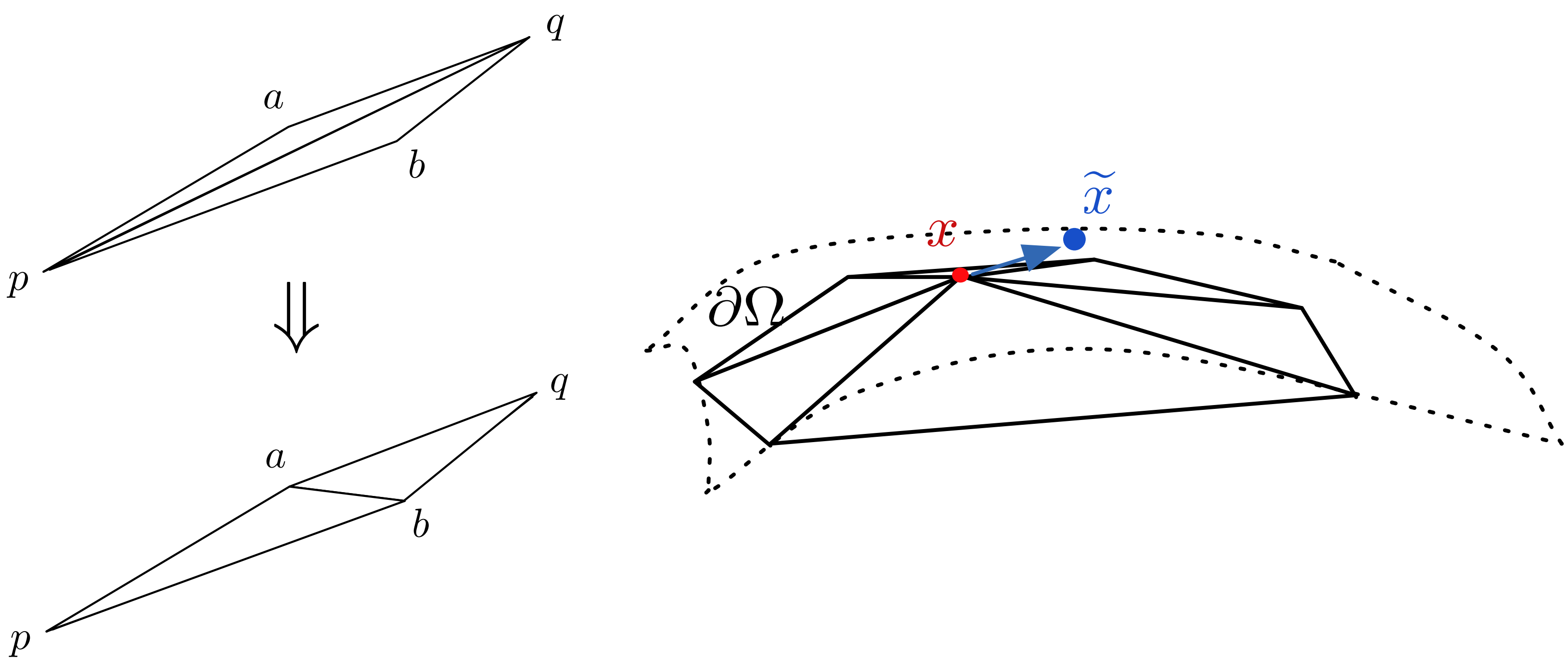


Figure 8: (left) $2d$ swap of edge pq, creating edge ab ; (right) relocation of node x to $\widetilde{x}$, along the surface.

Local remeshing in $3d$: numerical examples

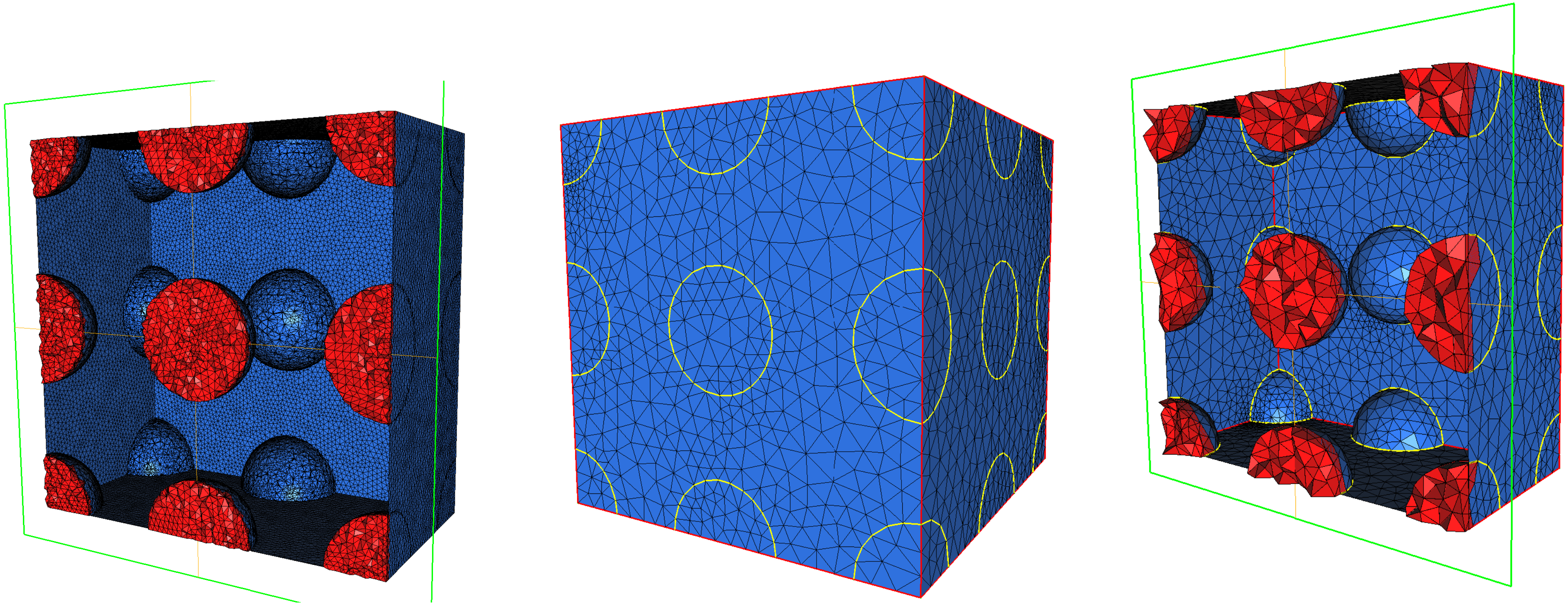


Figure 9: *(left) Ill-shaped discretization of an implicit function in a cube, (middle-right) result after local remeshing.*

Outline

I. Mathematical modeling of shape optimization problems
   1. Differentiation with respect to the domain : Hadamard's method
   2. The 'classical' level set method, and the proposed strategy

II. From meshed domains to a level set description,... and conversely
   1. Initializing level-set functions with the signed distance function
   2. Meshing the negative subdomain of a level set function : local remeshing

III. Application to shape optimization
   1. Numerical Implementation
   2. The algorithm in motion
   3. Some numerical results

Numerical implementation

- At each iteration, the shape Ω^n is endowed with an unstructured mesh $\mathcal{T}^n$ of a larger, fixed, bounding box $\mathcal{D}$, in which a mesh of Ω^n explicitly appears as a submesh.

- When computing the descent direction $\theta^n := -J\prime(\Omega^n)$, finite element computations are held on the sole Ω^n (the part of $\mathcal{T}^n$, exterior to Ω^n is simply 'forgotten')

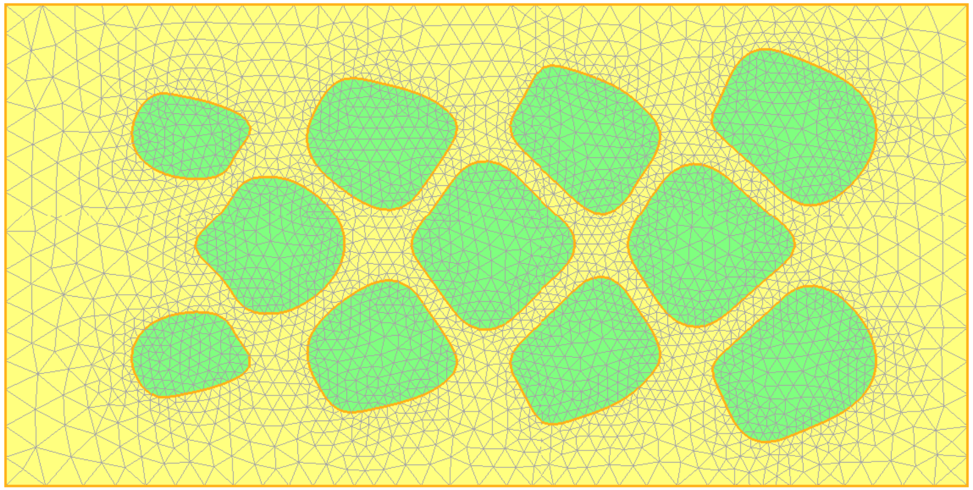
- When dealing with the advection step $\Omega^n \to \Omega^{n+1}$, the signed distance function d_{Ω^n} to Ω^n is generated on the whole mesh $\mathcal{T}^n$, and accounts for Ω^n in the resolution of the level set advection equation to get ϕ^{n+1}:

$$\begin{cases} \frac{\partial \phi}{\partial t}(t,x) + \theta^n(x).\nabla\phi(t,x) = 0 & \text{on } [0,\tau^n] \times \mathcal{D} \\ \phi(t=0,x) = d_{\Omega^n}(x) & \text{on } \mathcal{D} \end{cases}$$

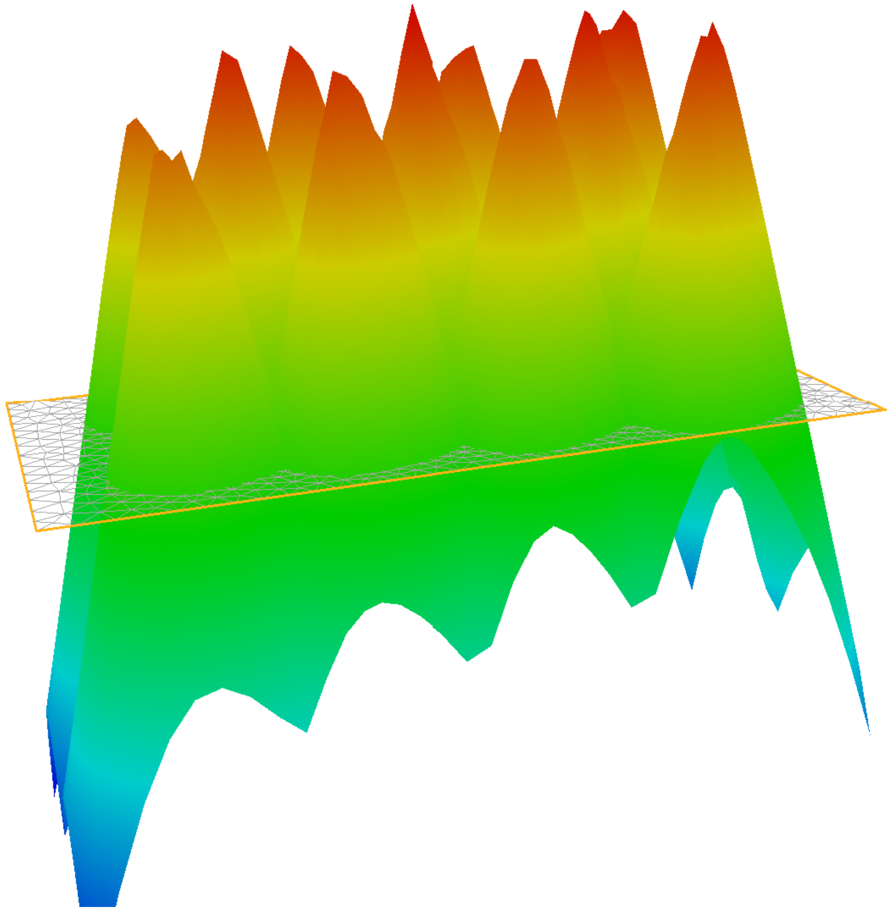
- From the knowledge of ϕ^{n+1}, a new unstructured mesh $\mathcal{T}^{n+1}$, in which the new shape Ω^{n+1} explicitly appears, is recovered, using the previous meshing techniques.

The algorithm in motion...

Step 1: Start with shape Ω^n, and generate its signed distance function d_{Ω^n} over $\mathcal{D}$, equipped with an unstructured mesh $\mathcal{T}^n$.



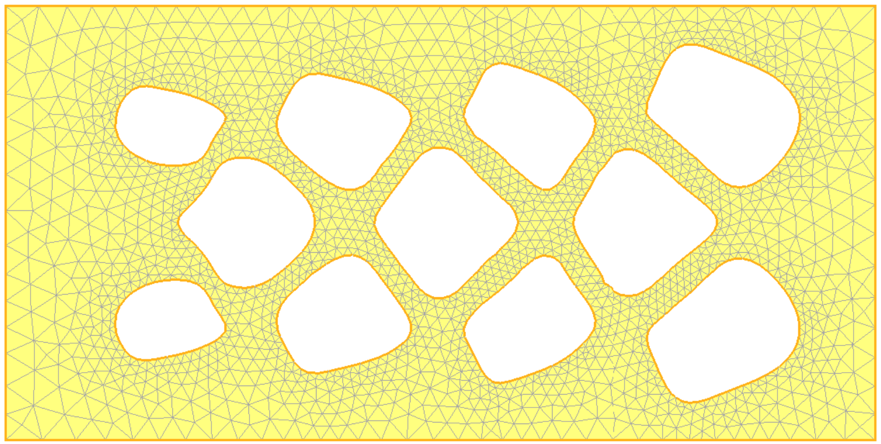
(a) The initial shape



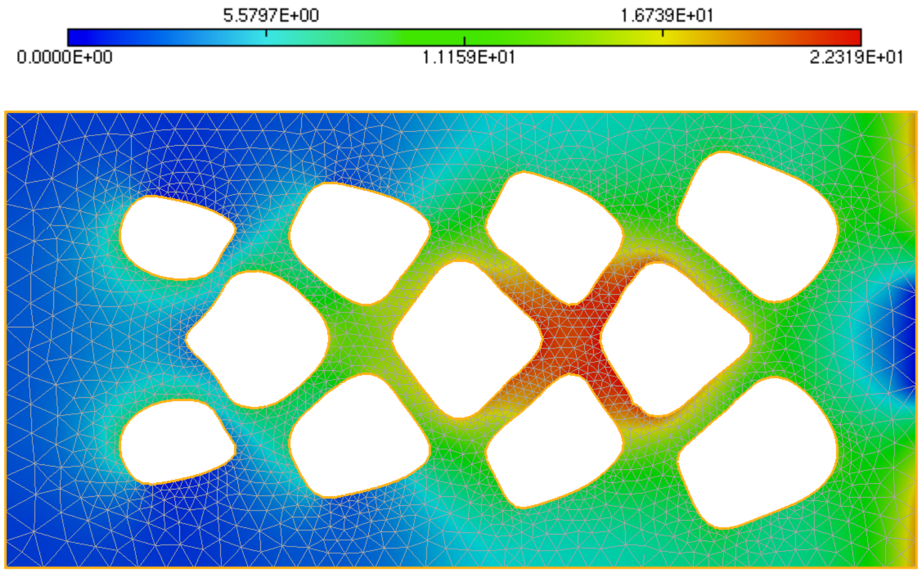
(b) Isolines of d_{Ω^n}

The algorithm in motion...

Step 2: "Forget" the exterior of the shape $\mathcal{D} \setminus \Omega^n$, and perform the computation of the shape gradient $J\prime(\Omega^n)$ on the shape.



(a) The "interior mesh"



(b) Computation of $J\prime(\Omega^n)$

The algorithm in motion...

Step 3: "Remember" the whole computational mesh $\mathcal{T}^n$. Extend the velocity field $J\prime(\Omega^n)$ to the whole mesh, and advect d_{Ω^n} along $J\prime(\Omega^n)$ for a (small) time step τ^n. A new level set function ϕ^{n+1} is obtained on $\mathcal{T}^n$, corresponding to the new shape Ω^{n+1}.

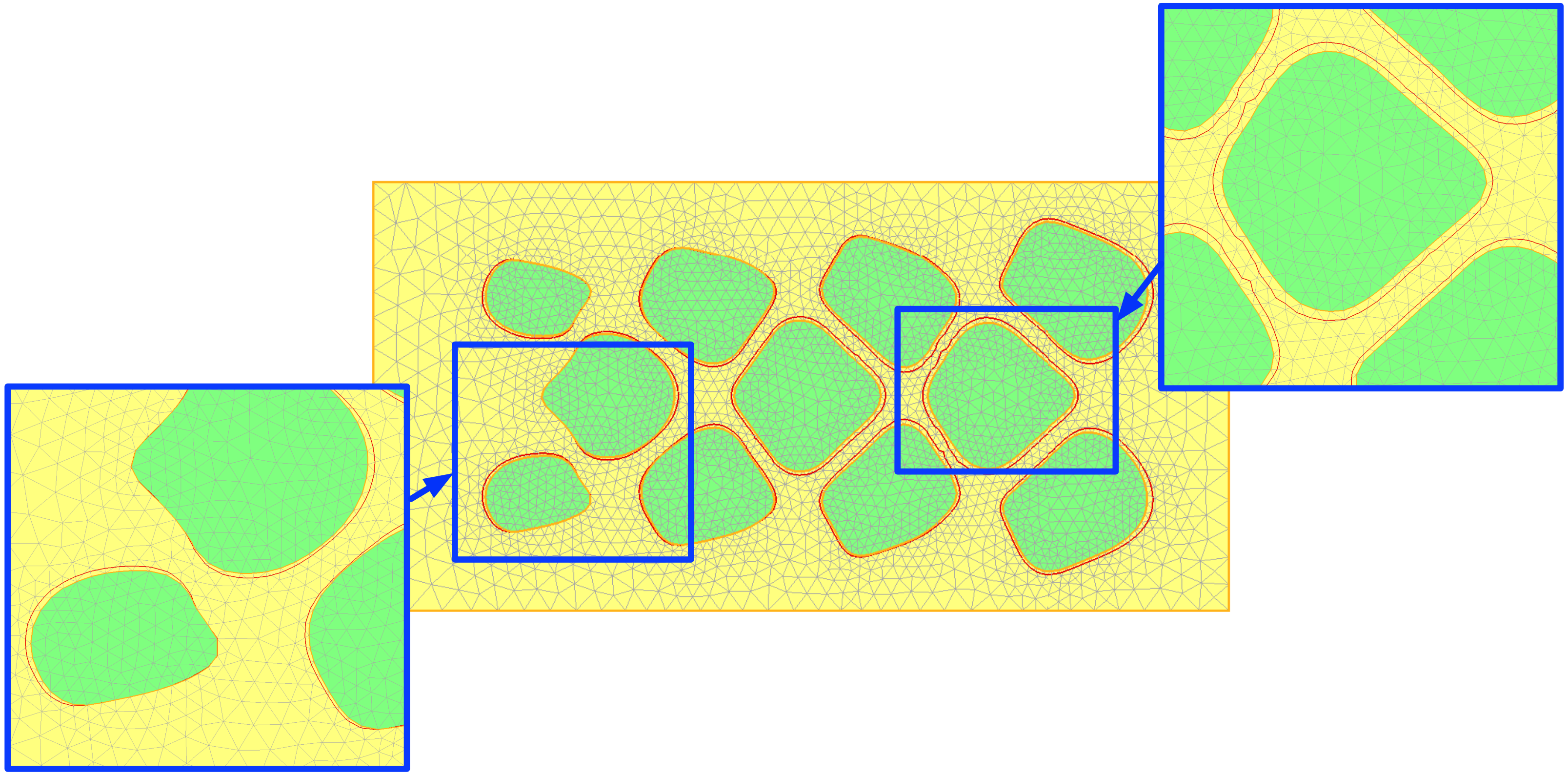


Figure 10: *The shape Ω^n, discretized in the mesh (in yellow), and the "new", advected 0-level set (in red).*

The algorithm in motion...

Step 4: To close the loop, the 0 level set of ϕ^{n+1} is explicitly discretized in mesh $\mathcal{T}^n$. As expected, roughly "breaking" this line generally yields a very ill-shaped mesh.

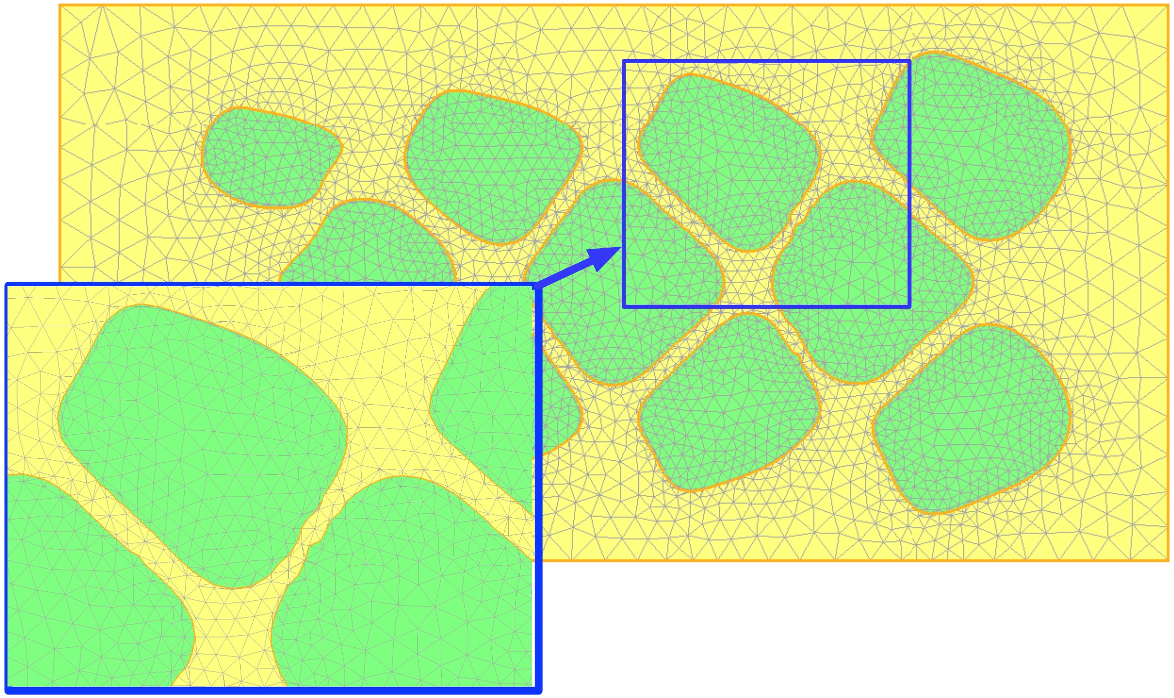


Figure 11: *Rough discretization of the* 0 *level set of* ϕ^{n+1} *into* $\mathcal{T}^n$*; the resulting mesh of* D *is ill-shaped.*

The algorithm in motion...

The mesh modification step is then performed, so as to enhance the overall quality of the mesh according to the geometry of the shape. $\mathcal{T}^{n+1}$ is eventually obtained.

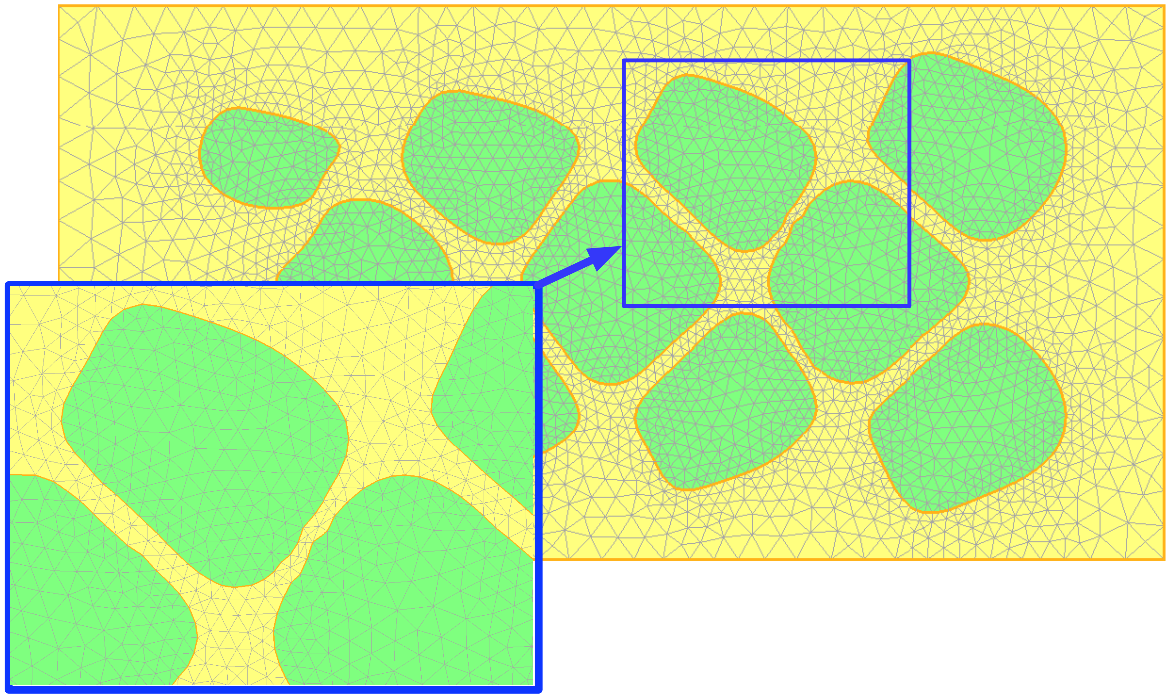
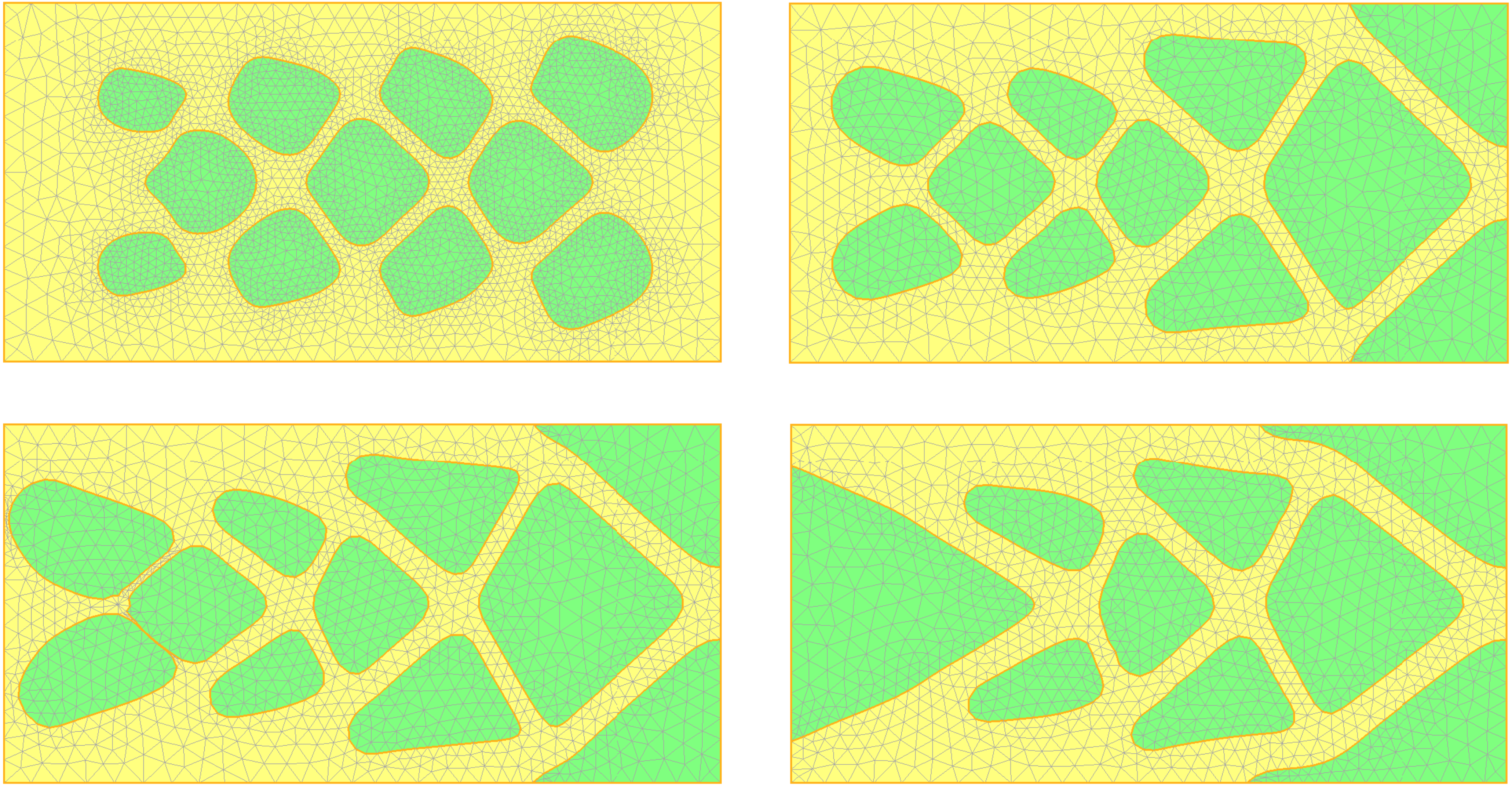


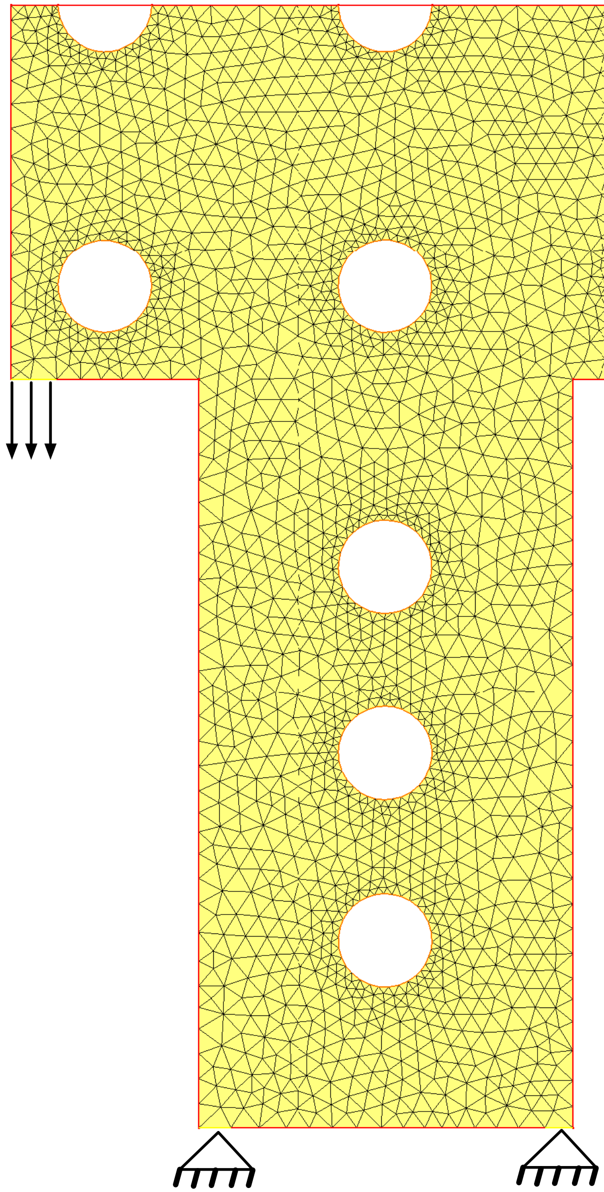
Figure 12: *Quality-oriented remeshing of the previous mesh ends with the new, well-shaped mesh $\mathcal{T}^{n+1}$ of $\mathcal{D}$ in which Ω^{n+1} is explicitly discretized.*

The algorithm in motion...

Go on as before, until convergence (discretize the 0-level set in the computational mesh, clean the mesh,...).



Some numerical results



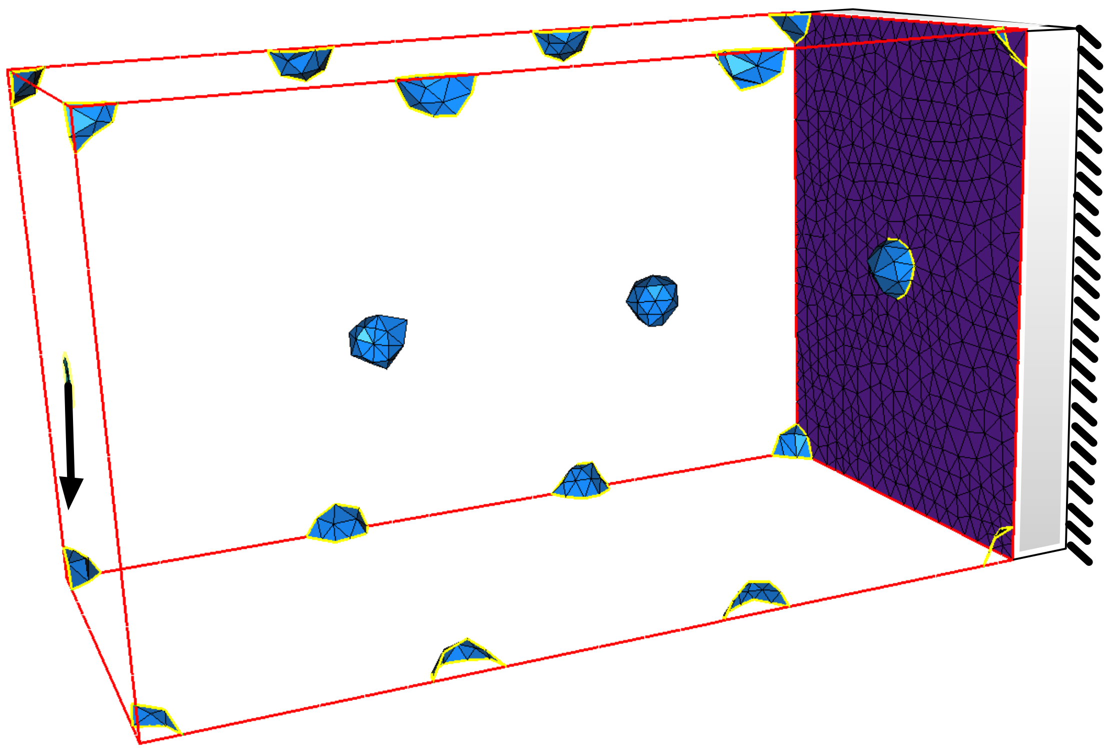
The 'benchmark' two-dimensional optimal mast test case.

- Minimization of the compliance

$$C(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega)\, dx.$$

- A volume constraint is enforced by means of a fixed Lagrange multiplier.

Some numerical results



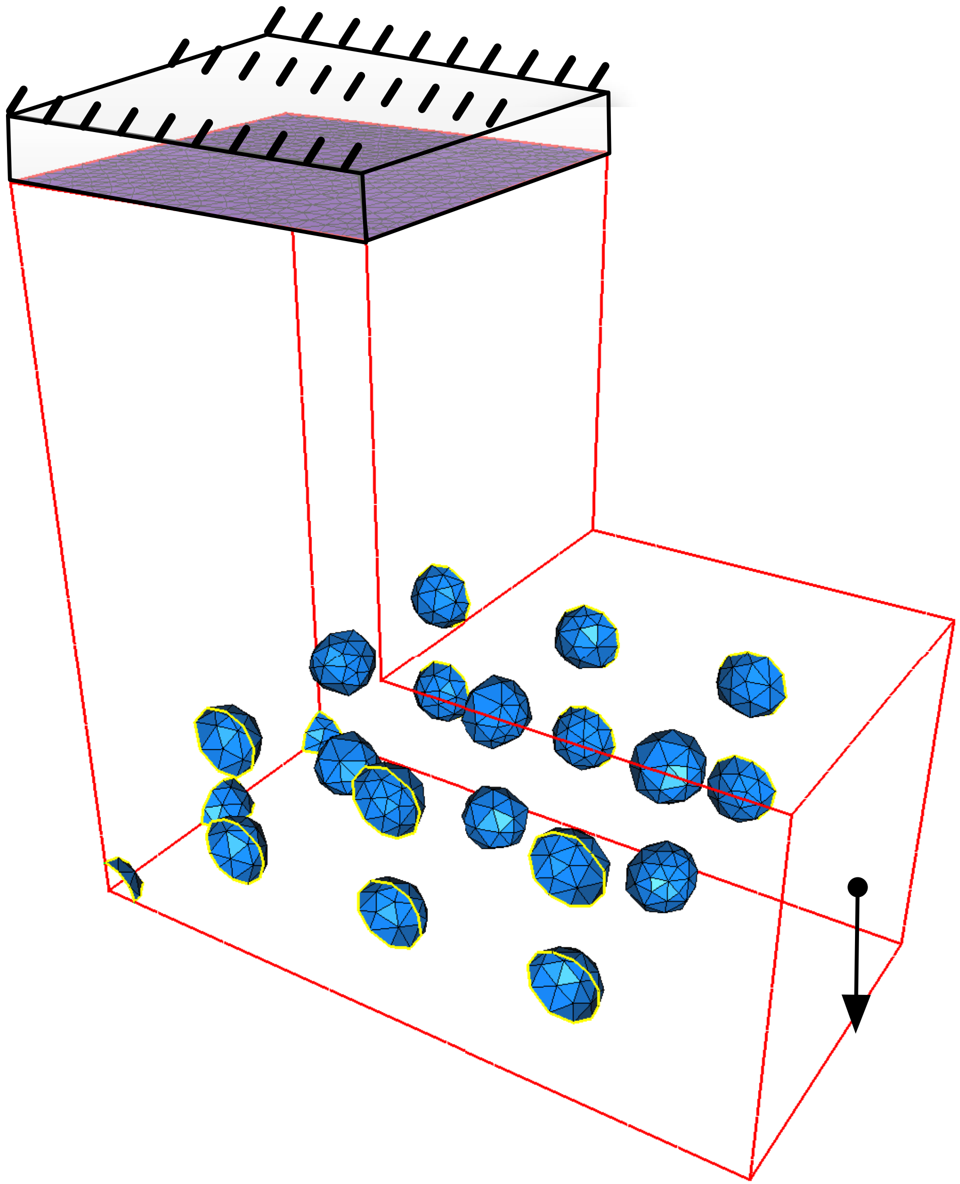
The 'benchmark' three-dimensional cantilever test case.

- Minimization of the compliance

$$C(\Omega) = \int_\Omega Ae(u_\Omega) : e(u_\Omega)\, dx.$$

- A volume constraint is enforced by means of a fixed Lagrange multiplier.

Some numerical results



The three-dimensional L-Beam test case.

- Minimization of a stress-based criterion

$$S(\Omega) = \int_\Omega k(x)||\sigma(u_\Omega)||^2 \, dx,$$

  where k is a weight factor, equal to 1 everywhere except in a neighborhood of the point where the load is exerted.
- A volume constraint is enforced by means of a fixed Lagrange multiplier.

Thank you !