front tracking shape optimization with a level set-based mesh evolution algorithm

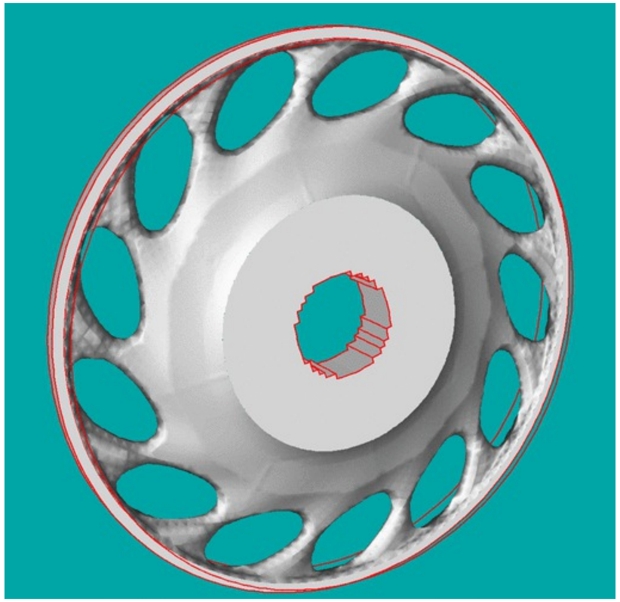
G. Allaire[1], Ch. Dapogny[1,2,3], and P. Frey[2]

[1] CMAP, UMR 7641 École Polytechnique, Palaiseau, France

[2] Laboratoire J.L. Lions, UPMC, Paris, France

[3] Technocentre Renault, Guyancourt

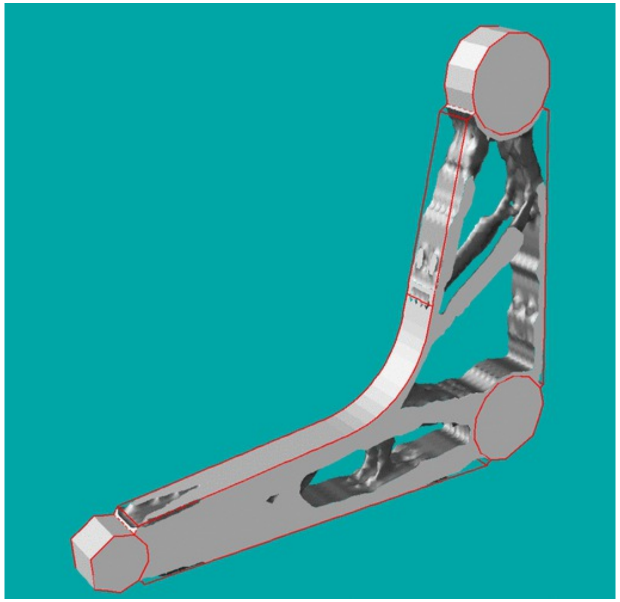
Shape optimization and industrial applications



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

Such problems are difficult, partly because

- they feature a very high computational cost, mainly due to repeated mechanical analyses.
- they require an accurate description of the various shapes that could be obtained through the optimization process.

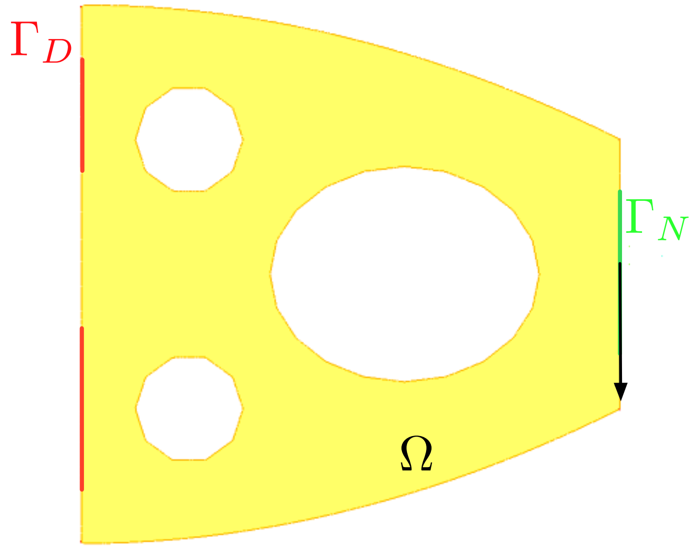


Automatic techniques (implemented in industrial softwares) have started to replace the traditional trial-and-error methods used by engineers, but still leave room for many forthcoming developments.

A model problem in linear elasticity

A shape is a bounded open domain $\Omega \subset \mathbb{R}^d$, which is

- fixed on a part $\Gamma_D \subset \partial\Omega$ of its boundary,
- submitted to surface loads g, applied on $\Gamma_N \subset \partial\Omega$, $\Gamma_D \cap \Gamma_N = \emptyset$.



A 'Cantilever'

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

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

where $e(u) = \frac{1}{2}(\nabla u^T + \nabla u)$ is the strain tensor field, and A is the Hooke's law of the material.

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 least-square discrepancy between the displacement u_Ω and a target displacement $u_0 \in H^1(\Omega)^d$ (useful when designing micro-mechanisms):

$$D(\Omega) = \left(\int_\Omega k(x) ||u_\Omega - u_0||^\alpha dx \right)^{\frac{1}{\alpha}},$$

where α is a fixed parameter, and $k(x)$ is a weight factor.

A volume constraint may be enforced with a fixed penalty parameter ℓ:

$$\text{Minimize } J(\Omega) := C(\Omega) + \ell \, \text{Vol}(\Omega), \text{ or } D(\Omega) + \ell \, \text{Vol}(\Omega).$$

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain: Hadamard's method
2. Numerical implementation of shape optimization algorithms
3. The proposed method

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

1. A few words about the level set Method
2. Initializing level set functions with the signed distance function
3. 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. Numerical implementation of shape optimization algorithms
3. The proposed method

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

1. A few words about the level set Method
2. Initializing level-set functions with the signed distance function
3. Meshing the negative subdomain of a level set function: local remeshing

III. Application to shape optimization

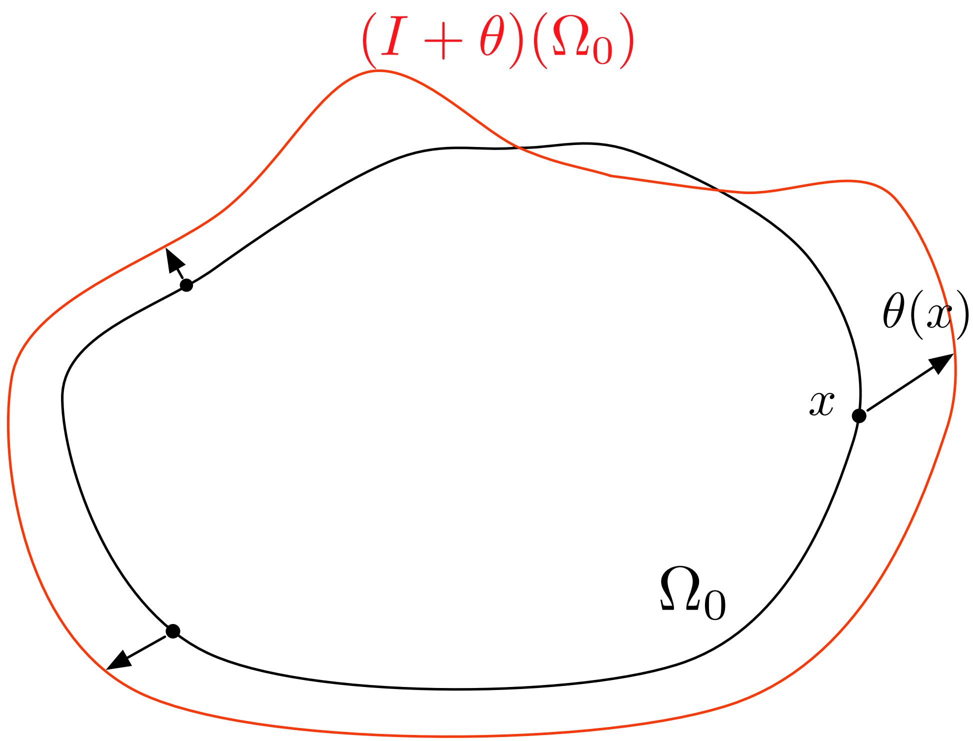
1. Numerical implementation
2. The algorithm in motion
3. Some numerical results

Differentiation with respect to the domain: Hadamard's method

Hadamard's boundary variation method describes variations of a reference, Lipschitz domain Ω_0 of the form:

$$\Omega_0 \to (I+\theta)(\Omega_0),$$

for 'small' $\theta \in W^{1,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right)$.



Lemma 1 *For all $\theta \in W^{1,\infty}\left(\mathbb{R}^d, \mathbb{R}^d\right)$ with norm $||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)} < 1$, $(I+\theta)$ is a Lipschitz diffeomorphism of $\mathbb{R}^d$, with Lipschitz inverse.*

Differentiation with respect to the domain: Hadamard's method

Definition 1 *Given a smooth domain Ω_0, a (scalar) function $\Omega \mapsto F(\Omega)$ is shape differentiable at Ω_0 if the function*

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

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

$$F((I+\theta)(\Omega_0)) = F(\Omega_0) + F'(\Omega_0)(\theta) + o\left(||\theta||_{W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)}\right).$$

Techniques close to optimal control theory make it possible to compute shape gradients; in the case of 'many' functionals of the domain $J(\Omega)$, the shape derivative has the particular structure:

$$J'(\Omega)(\theta) = \int_\Gamma v\, \theta \cdot n\, ds,$$

where v is a scalar field which depends on u_Ω, and possibly on an adjoint state p_Ω.

Example: If $J(\Omega) = C(\Omega) = \int_{\Gamma_N} g \cdot u_\Omega\, ds$ is the compliance, $v = -Ae(u_\Omega) : e(u_\Omega)$.

Differentiation with respect to the domain: Hadamard's method

- This shape gradient provides a natural descent direction for functional J: *for instance*, defining θ as

$$\theta = -vn$$

yields, for $t > 0$ sufficiently small (*to be found numerically*):

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

- Hadamard's method suffers several drawbacks (dependence on the initialization, non existence of global minimizer, etc...) which can be alleviated by using concurrent methods:

1. Topological gradient algorithms assess the sensitivity of shapes with respect to the nucleation of small holes.
2. The homogenization method is a relaxation of the minimization problem that provides a method for finding the global minimum of the relaxed problem.

The generic numerical algorithm

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

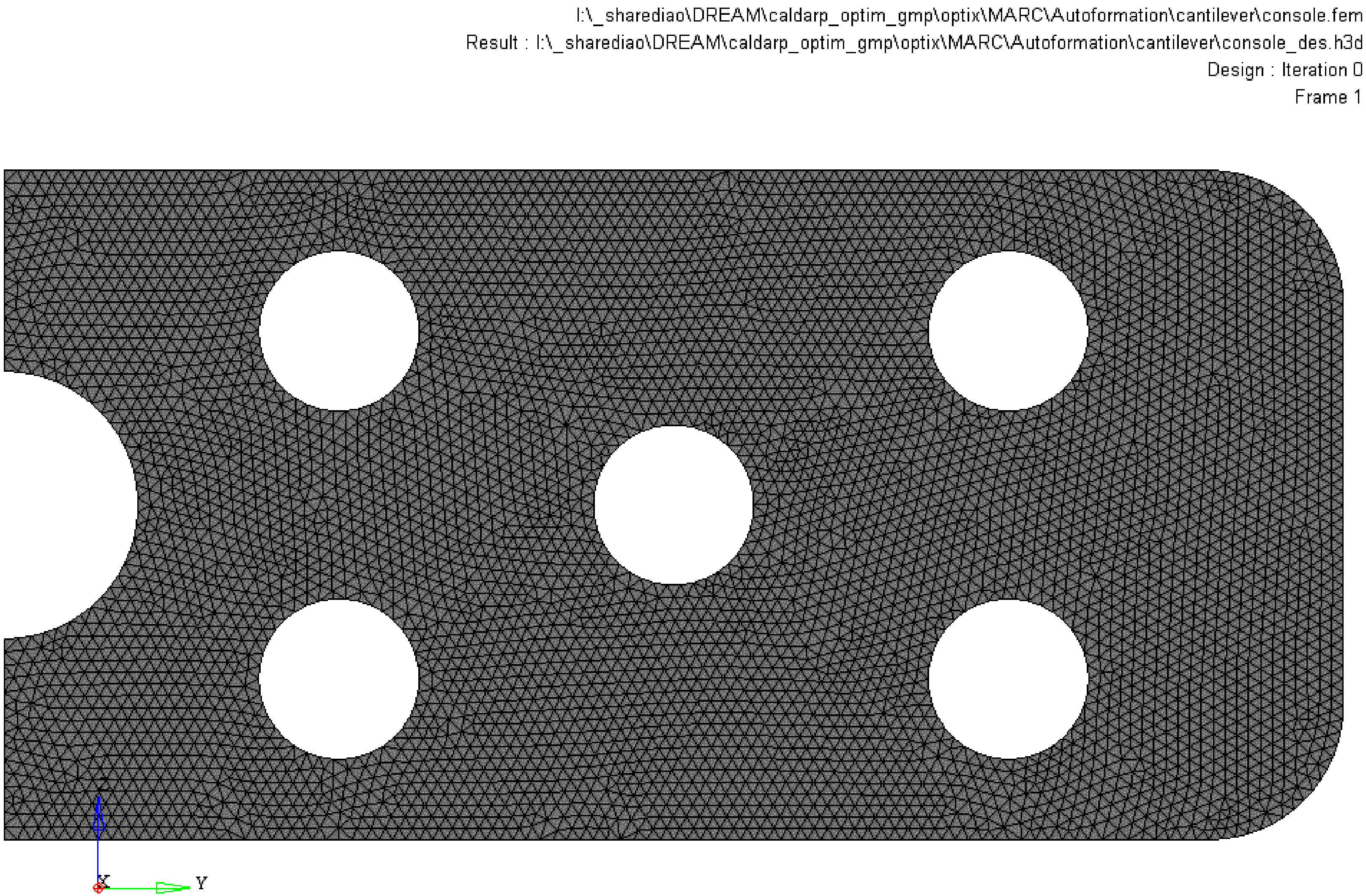
1. Compute the solution u_{Ω^n} of the above elasticity system of Ω^n.
2. Compute the shape gradient $J'(\Omega^n)$ thanks to the previous formula, and infer a descent direction θ^n for the cost functional.
3. Advect the shape Ω^n according to this displacement field, so as to get Ω^{n+1}.

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.

The generic numerical algorithm

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



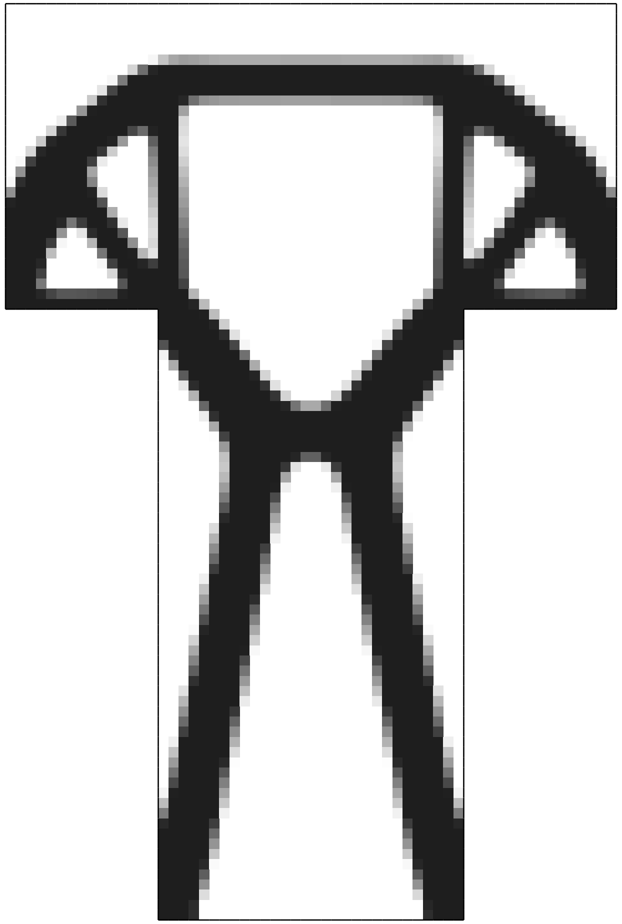
The level set method of Allaire-Jouve-Toader

- The shapes Ω^n are embedded in a computational box 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 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 $D \setminus \Omega^n$ is filled with a very 'soft' material, which leads to an approximate linear elasticity system, defined on 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

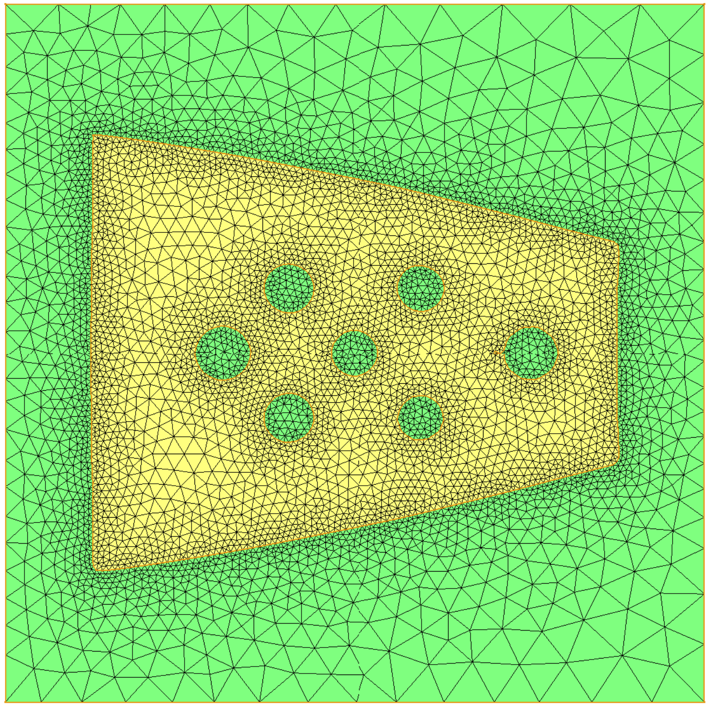
The proposed method

- still benefits from the versatility of level set methods to account for large deformations of shapes (even topological changes)
- yet, it enjoys at each step the knowledge of a mesh of the shape.

The computational box D is equipped with an unstructured mesh $\mathcal{T}^n$, which changes at each step n, so that the shape Ω^n is explicitly discretized in it.

- 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. Numerical implementation of shape optimization algorithms
   3. The proposed method

II. From meshed domains to a level set description,... and conversely
   1. A few words about the level set Method
   2. Initializing level-set functions with the signed distance function
   3. 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: [Osher & Sethian, 1988] *the motion of an evolving domain is best described in an implicit way.*

A bounded domain $\Omega \subset \mathbb{R}^d$ is equivalently defined by a function $\phi : \mathbb{R}^d \to \mathbb{R}$ such that:

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

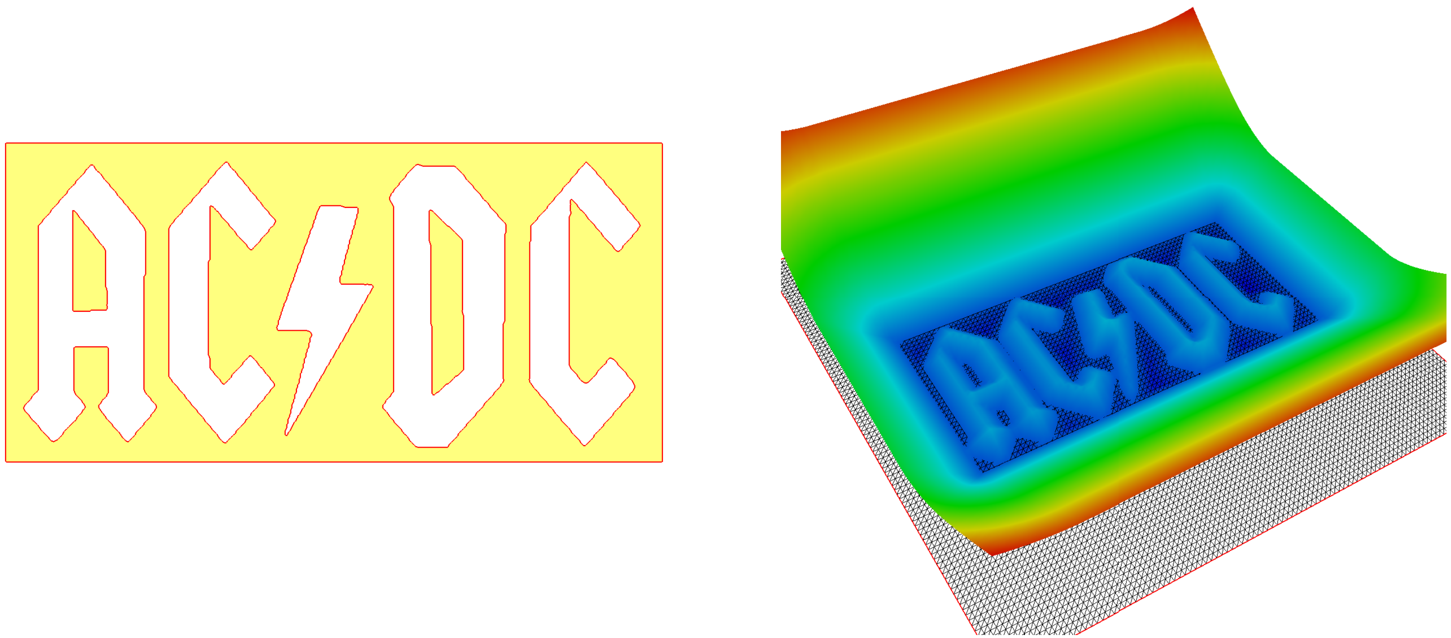


Figure 1: *A bounded domain $\Omega \subset \mathbb{R}^2$ (left), some level sets of an associated level set function (right).*

Surface evolution equations in the level set framework

The motion of an evolving domain $\Omega(t) \subset \mathbb{R}^d$ along a velocity field $v(t,x) \in \mathbb{R}^d$ is translated in terms of an associated 'level set function' $\phi(t,.)$ by the level set advection equation:

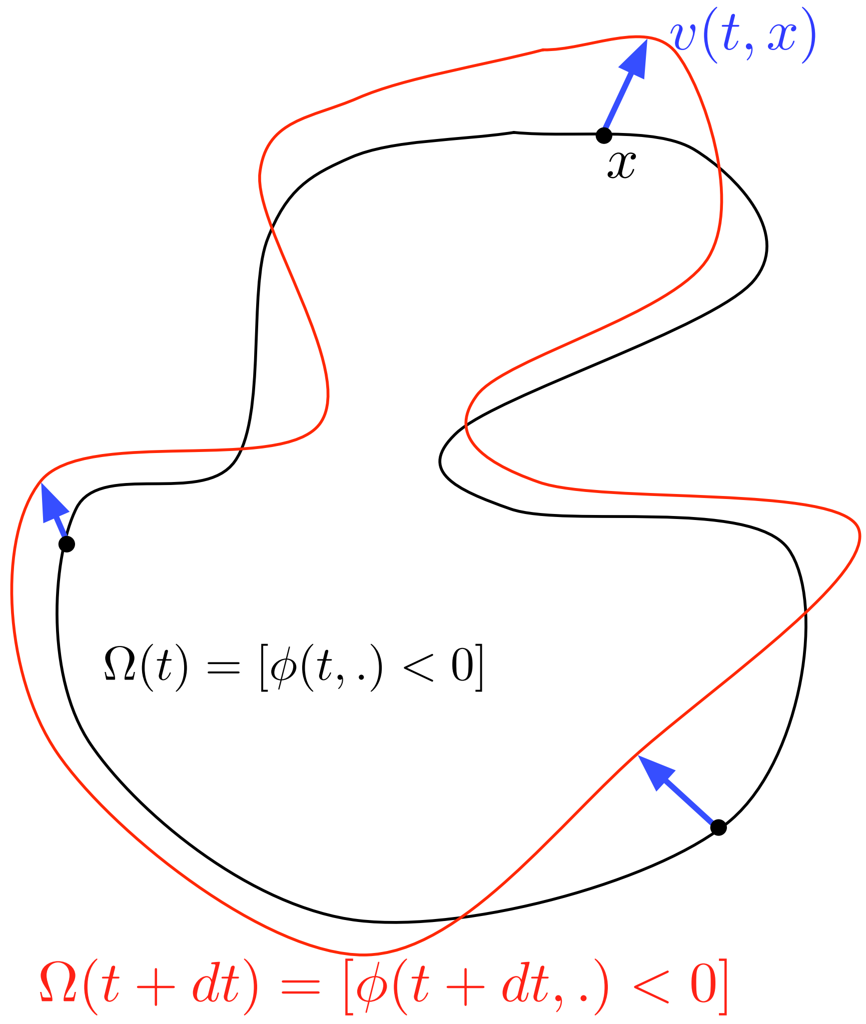
$$\forall t,\ \forall x \in \mathbb{R}^d,\ \frac{\partial \phi}{\partial t}(t,x) + v(t,x).\nabla\phi(t,x) = 0$$

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 equation:

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



Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain: Hadamard's method
2. Numerical implementation of shape optimization algorithms
3. The proposed method

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

1. A few words about the level set Method
2. Initializing level-set functions with the signed distance function
3. 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 bounded domain. The signed distance function to Ω is the function $\mathbb{R}^d \ni x \mapsto d_\Omega(x)$ defined by:*

$$d_\Omega(x) = \begin{cases} -d(x, \partial\Omega) & \text{if } x \in \Omega \\ 0 & \text{if } x \in \partial\Omega \\ d(x, \partial\Omega) & \text{if } x \in \overline{{}^c\Omega} \end{cases}$$ *, where $d(\cdot, \partial\Omega)$ is the usual Euclidean distance*

- The signed distance function to a domain $\Omega \subset \mathbb{R}^d$ is the 'canonical' way to initialize an associated level set function, mainly owing to its unit gradient property:

$$||\nabla d_\Omega(x)|| = 1, \quad \text{p.p } x \in \mathbb{R}^d.$$

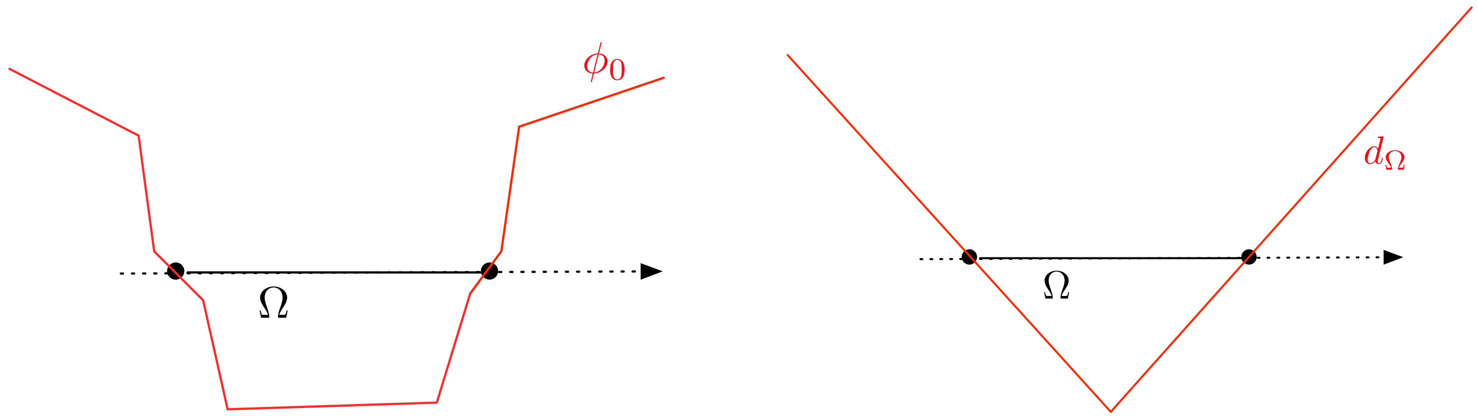


Figure 2: (left) Any level set function for $\Omega = (0,1) \subset \mathbb{R}$; (right) signed distance function to Ω.

The signed distance function as the steady state of a PDE

Suppose $\Omega \subset \mathbb{R}^d$ is implicitly known as

$$\Omega = \left\{x \in \mathbb{R}^d ; \phi_0(x) < 0\right\} \text{ and } \partial\Omega = \left\{x \in \mathbb{R}^d ; \phi_0(x) = 0\right\},$$

where ϕ_0 is a function we only suppose continuous. Then the function u_Ω can be considered as the steady state of the so-called unsteady Eikonal equation

$$\begin{cases} \dfrac{\partial \phi}{\partial t} + \mathrm{sgn}(\phi_0)(||\nabla \phi|| - 1) = 0 & \forall t > 0, x \in \mathbb{R}^d \\ \phi(t = 0, x) = \phi_0(x) & \forall x \in \mathbb{R}^d \end{cases} \tag{1}$$

More accurately,

Theorem 1 *[Aubert & Aujol, 2002] Define function ϕ, $\forall x \in \mathbb{R}^d$, $\forall t \in \mathbb{R}_+$,*

$$\phi(t,x) = \begin{cases} \mathrm{sgn}(\phi_0(x)) \inf\limits_{||y|| \leq t} (\mathrm{sgn}(\phi_0(x))\phi_0(x+y) + t) & \text{if } t \leq d(x, \partial\Omega) \\ \mathrm{sgn}(\phi_0(x)) d(x, \partial\Omega) & \text{if } t > d(x, \partial\Omega) \end{cases} \tag{2}$$

Let $T \in \mathbb{R}_+$. Then ϕ is the unique uniformly continuous viscosity solution of (1) such that, for all $0 \leq t \leq T$, $\phi(t,x) = 0$ on $\partial\Omega$.

The signed distance function as the steady state of a PDE

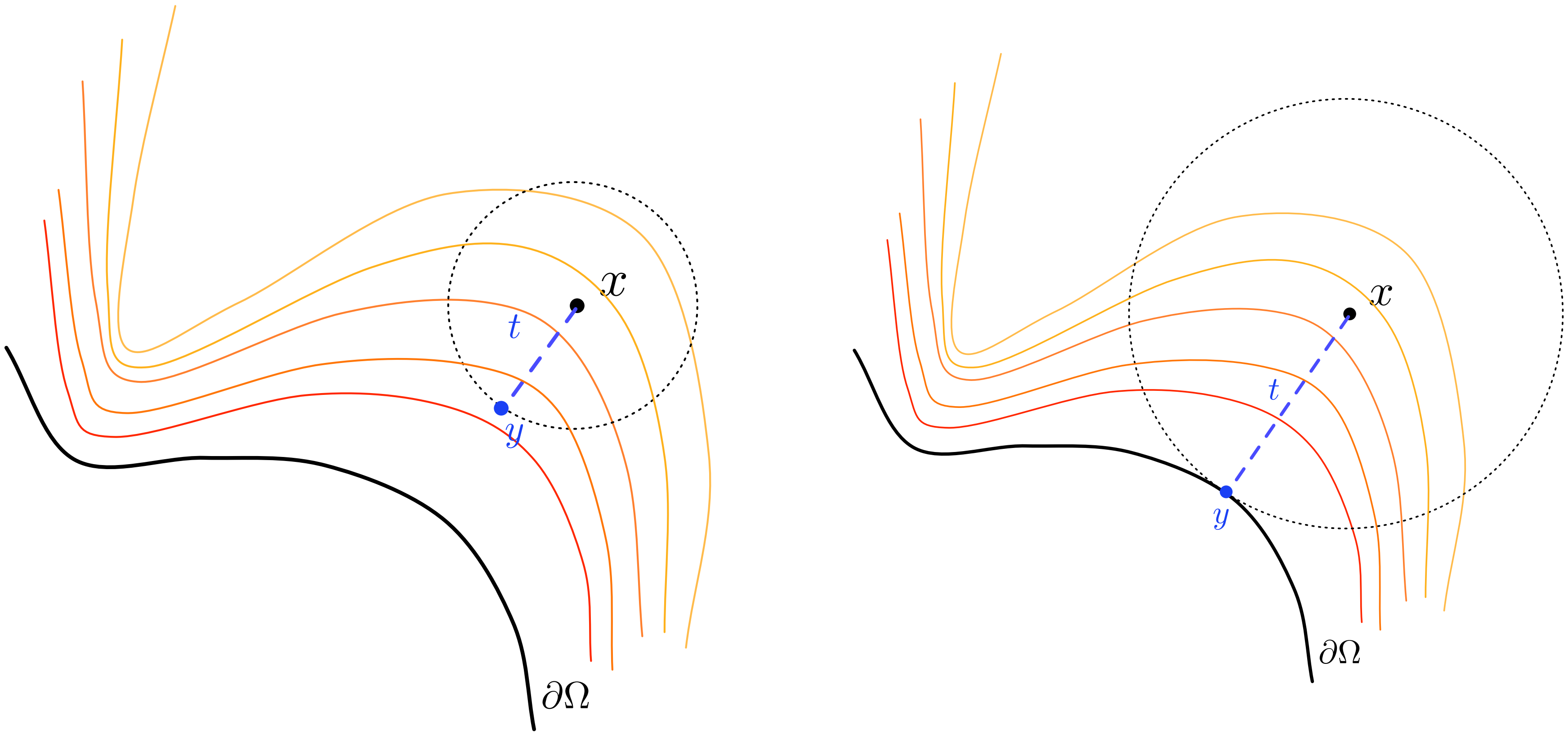


Figure 3: Some level sets of function ϕ_0; *(left):* computation of $\phi(t,x) = \phi_0(y) + t$ for small t; *(right):* computation of $\phi(t,x) = \phi_0(y) + t = d(x,\partial\Omega)$ at $t = d(x,\partial\Omega)$.

The proposed algorithm

Basic idea: *Compute iteratively the solution $\phi(t,x)$, using the exact formula.*

Let dt be a time step, and $t^n = ndt$.

The continuous formula for ϕ can be made iterative: denoting $\phi^n(x) = \phi(t^n, x)$, we have, for $n = 0, \dots$

$$\forall x \in {}^c\Omega, \ \phi^{n+1}(x) = \inf_{||y|| \leq dt} \phi^n(x+y) + dt$$

$$\forall x \in \Omega, \ \phi^{n+1}(x) = \sup_{||y|| \leq dt} \phi^n(x+y) - dt$$

and, dt being small enough, the above infimum and supremum are evaluated by taking y in the gradient direction; at a vertex x of the computational mesh $\mathcal{T}$:

$$\forall x \in {}^c\Omega, \ \phi^{n+1}(x) \approx \inf_{T \in Ball(x)} \phi^n \left(x - dt \frac{\nabla \phi^n|_T}{||\nabla \phi_n|_T||} \right) + dt$$

$$\forall x \in \Omega, \ \phi^{n+1}(x) \approx \sup_{T \in Ball(x)} \phi^n \left(x + dt \frac{\nabla \phi^n|_T}{||\nabla \phi_n|_T||} \right) - dt.$$

A 2d computational example

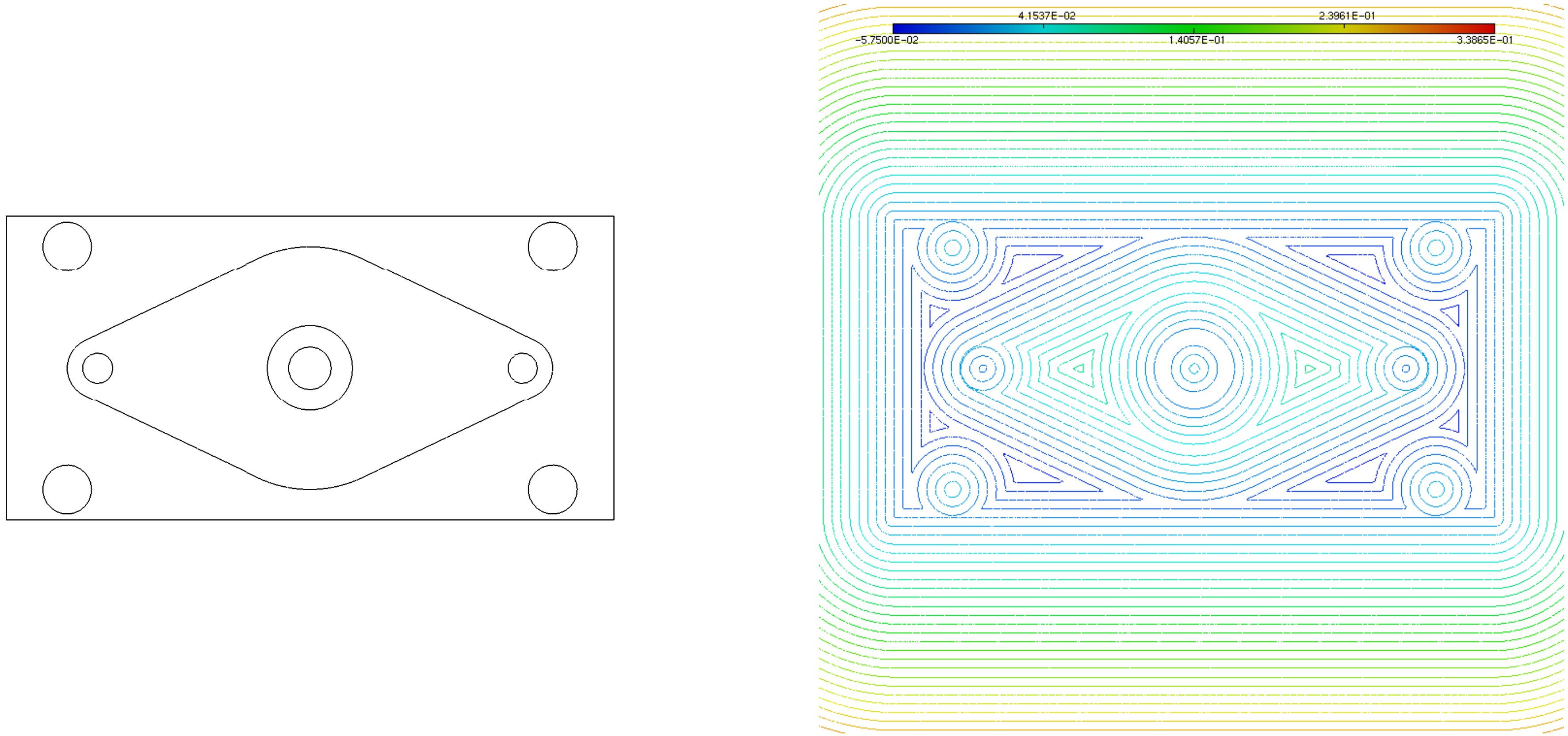


Figure 4: *Computation of the signed distance function to a discrete contour (left), on a fine background mesh (≈ 250000 vertices).*

A 3d example... the 'Aphrodite'.

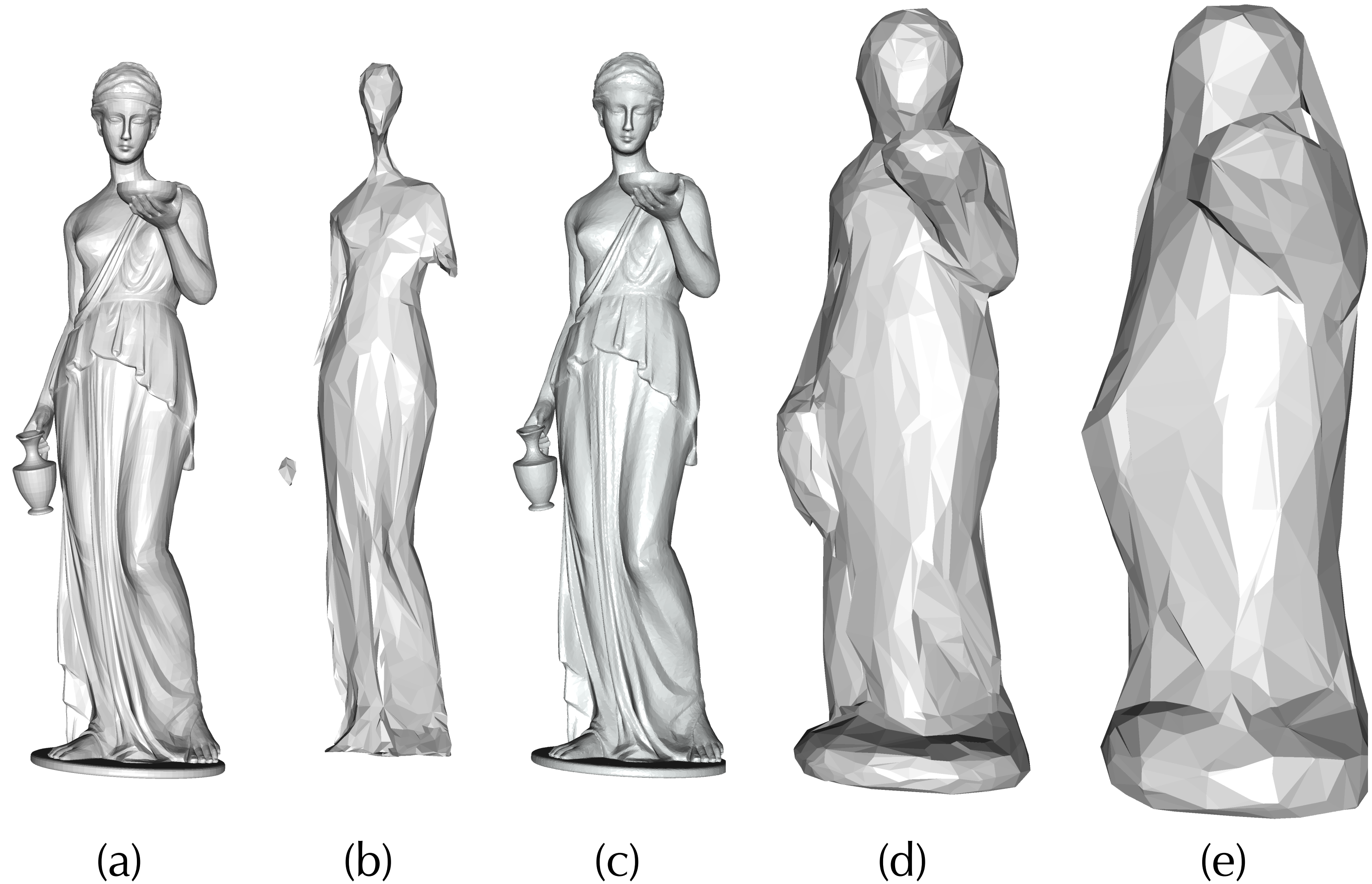


Figure 5: *Isosurfaces of the signed distance function to the 'Aphrodite' (a): (b): isosurface* -0.01*, (c): isosurface* 0*, (d): isosurface* 0.02*, (e): isosurface* 0.05.

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain: Hadamard's method
2. Numerical implementation of shape optimization algorithms
3. The proposed method

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

1. A few words about the level set Method
2. Initializing level-set functions with the signed distance function
3. 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 D is relatively easy, resorting to patterns.

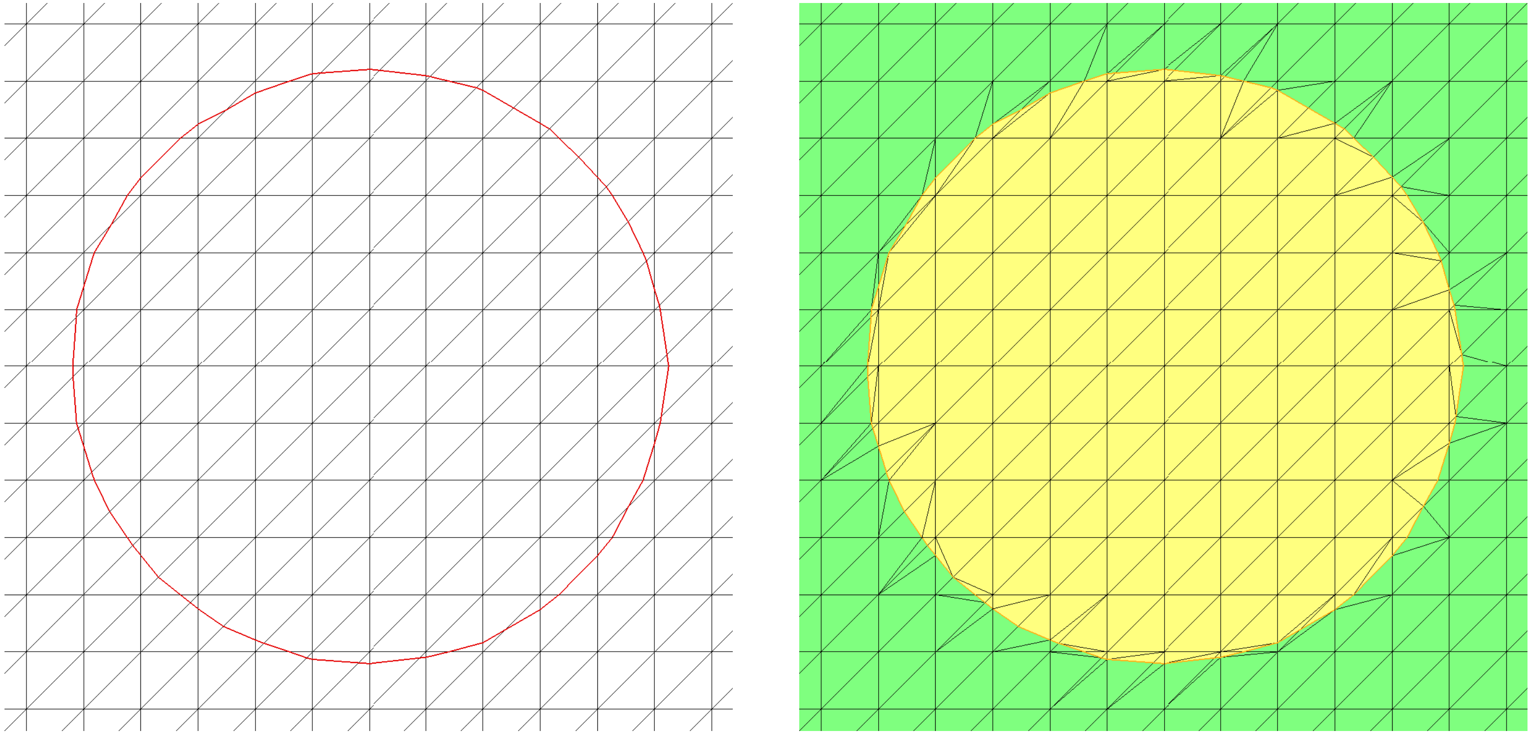


Figure 6: (left) 0 level set of a scalar function defined over a mesh ; (right) explicit discretization in the mesh.

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 $\mathcal{T}$ be 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 $\partial\Omega$.

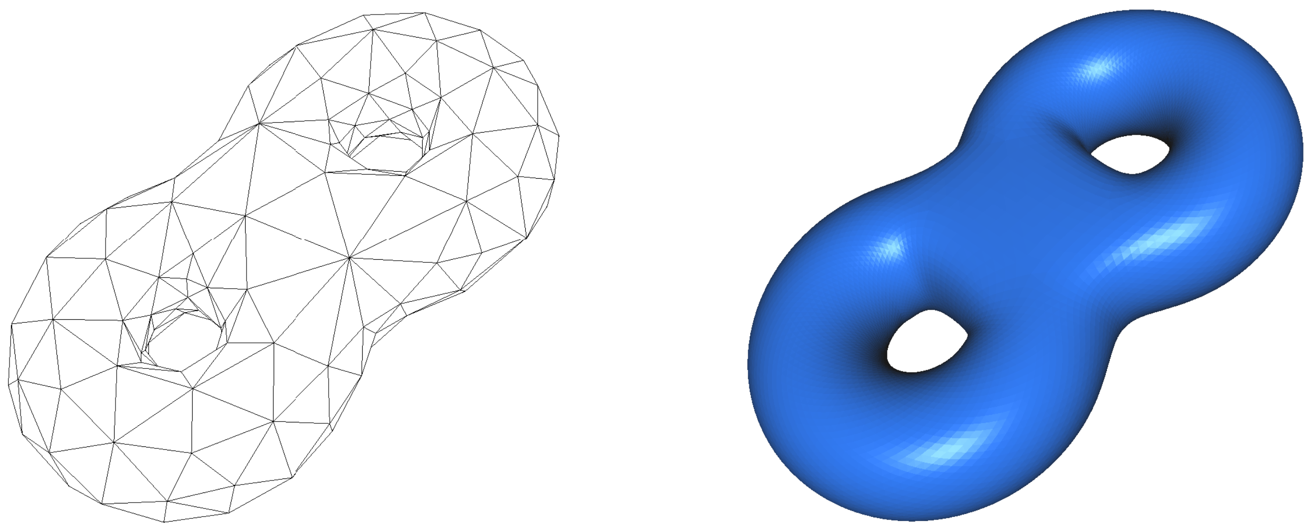


Figure 7: Poor geometric approximation (left) of a domain with smooth boundary (right)

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

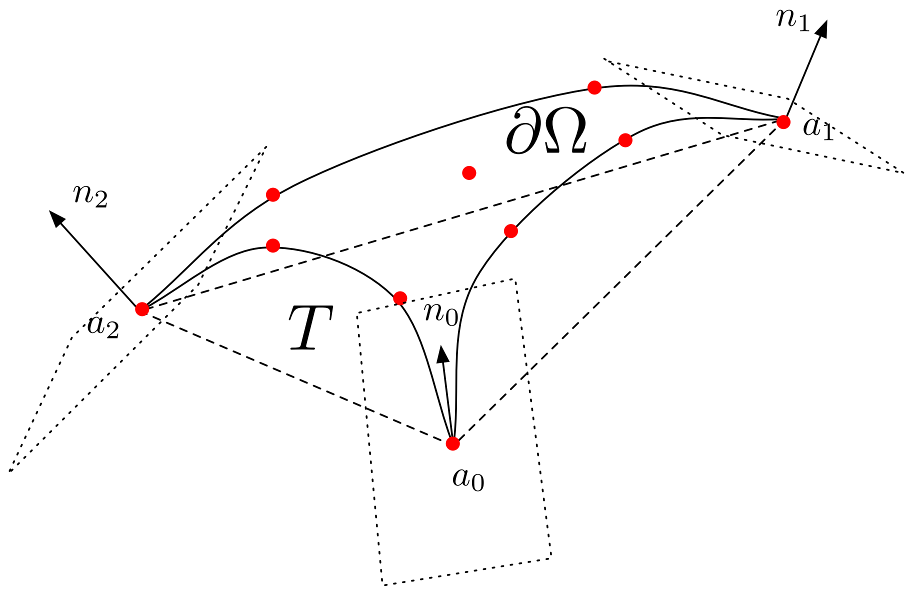


Figure 8: Generation of a cubic Bézier polynomial parametrization for the piece of $\partial\Omega$ associated to triangle T, from the approximated geometrical features (normal vectors at nodes).

Local remeshing in $3d$: remeshing strategy

- Four local remeshing operators are intertwined, to iteratively increase the quality of the mesh $\mathcal{T}$: edge split, edge collapse, edge swap, and vertex relocation.

- Each one of them exists under two different forms, depending on whether it is applied to a surface configuration, or an internal one.

- A size map h is defined, to reach a good mesh sampling. It generally depends on the principal curvatures κ_1, κ_2 of $\partial\Omega$, but may also be user-defined (e.g. in a context of mesh adaptation).

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 $\partial\Omega$,...

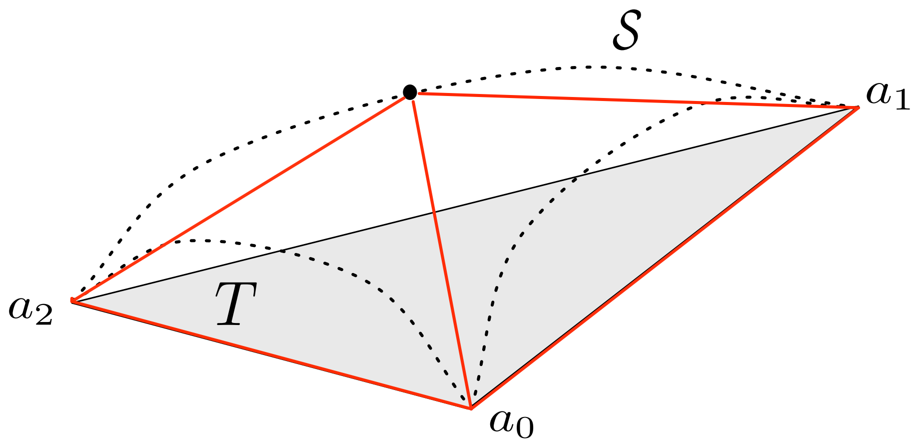


Figure 9: Splitting of one (left) or three (right) edges of triangle T, positioning the three new points on the ideal surface $\mathcal{S}$ (dotted).

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 long when compared to the prescribed size, or because it proves unnecessary to a nice geometric approximation of $\partial\Omega$,...

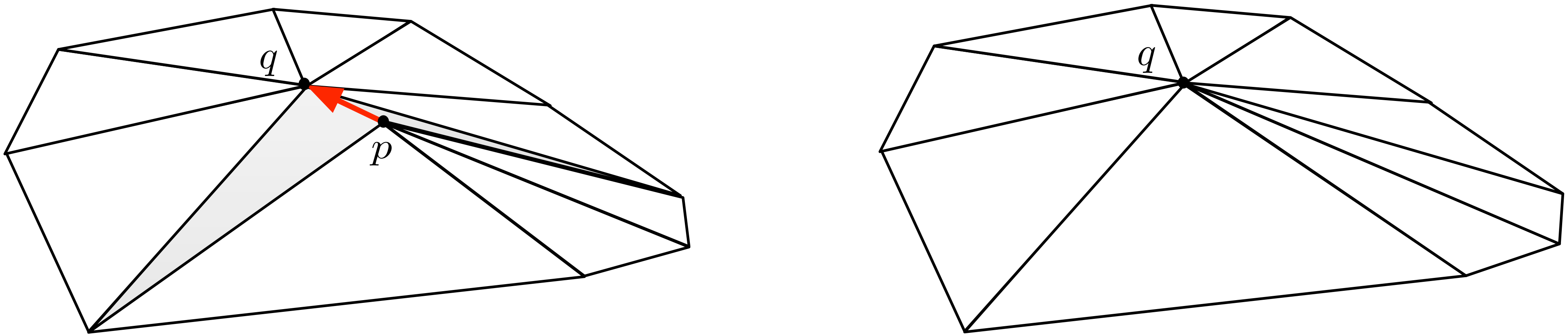


Figure 10: Collapse of point p over q.

Local mesh operators: edge collapse

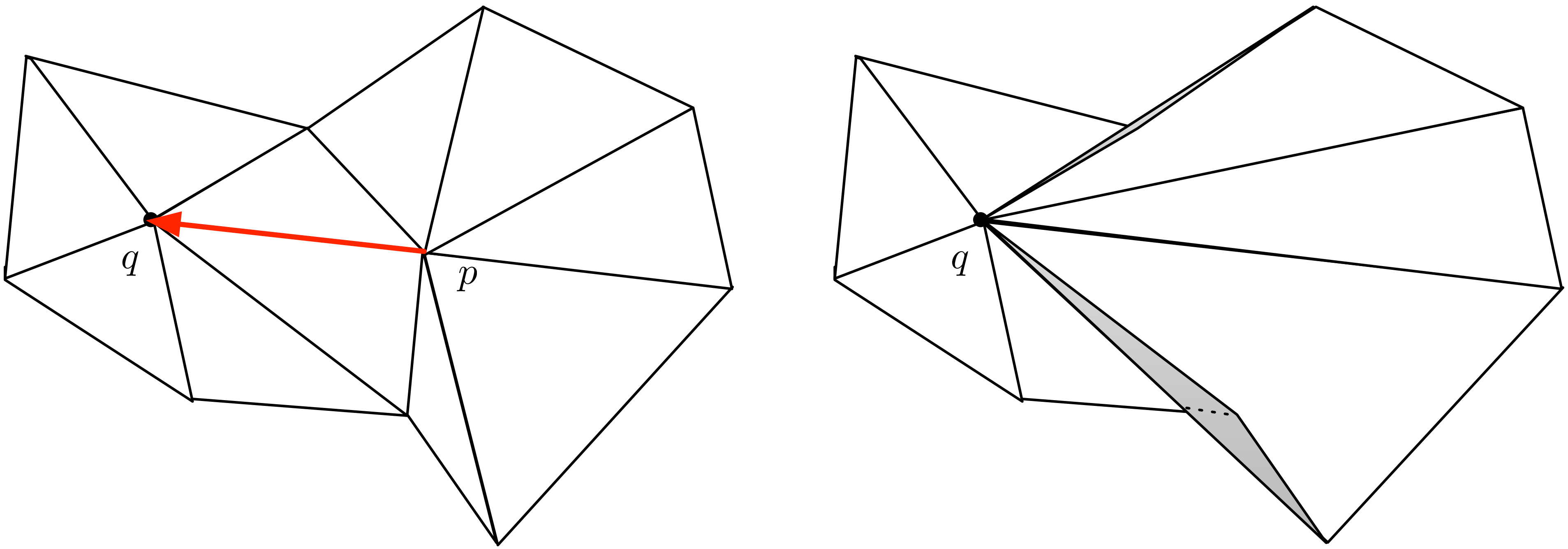


Figure 11: In two dimensions, collapsing p over q (left) invalidates the resulting mesh (right): both greyed triangles end up inverted.

Local mesh operators: edge swap, node relocation

For the sake of enhancement of the global quality of the mesh (or the geometrical approximation of $\partial\Omega$), some connectivities can be swapped, and some nodes can be slightly moved.

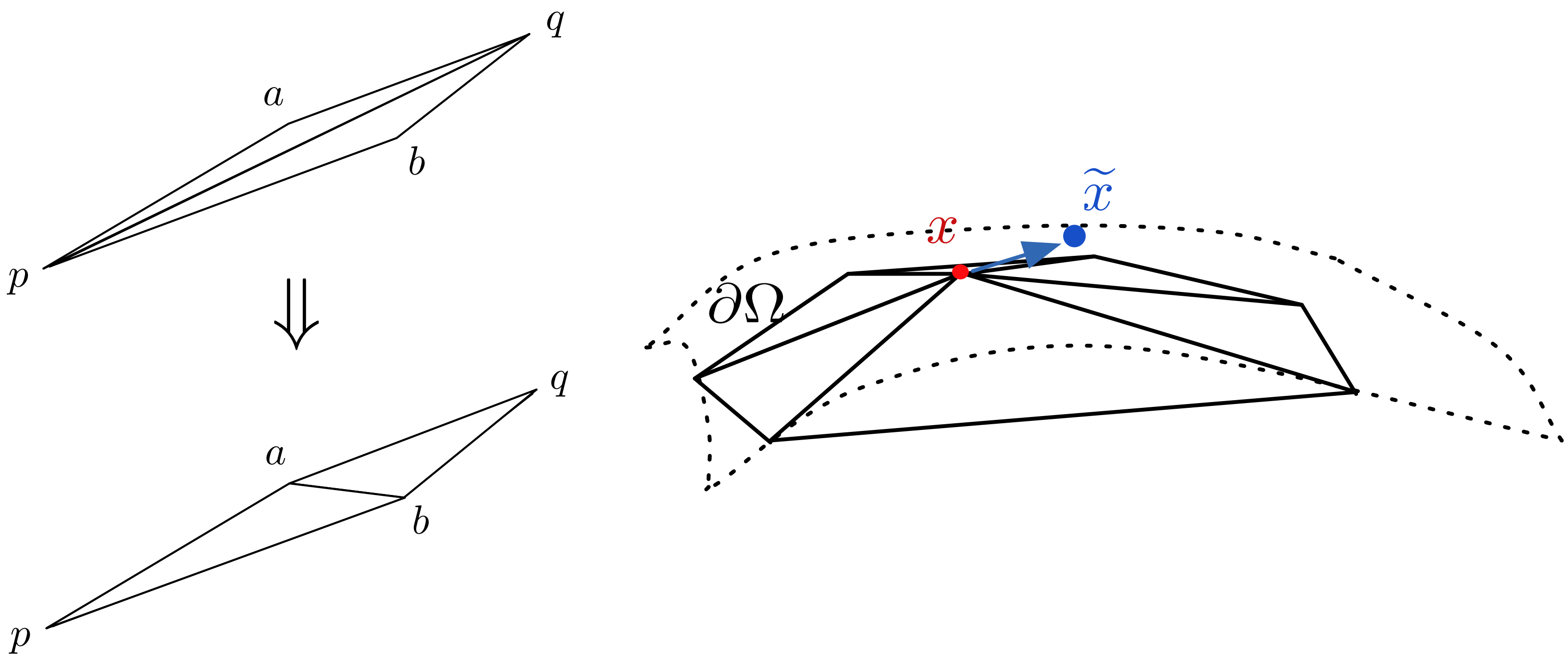


Figure 12: (left) $2d$ swap of edge pq, creating edge ab ; (right) relocation of node x to $\widetilde{x}$, along the surface.

Local remeshing in 3d: numerical examples

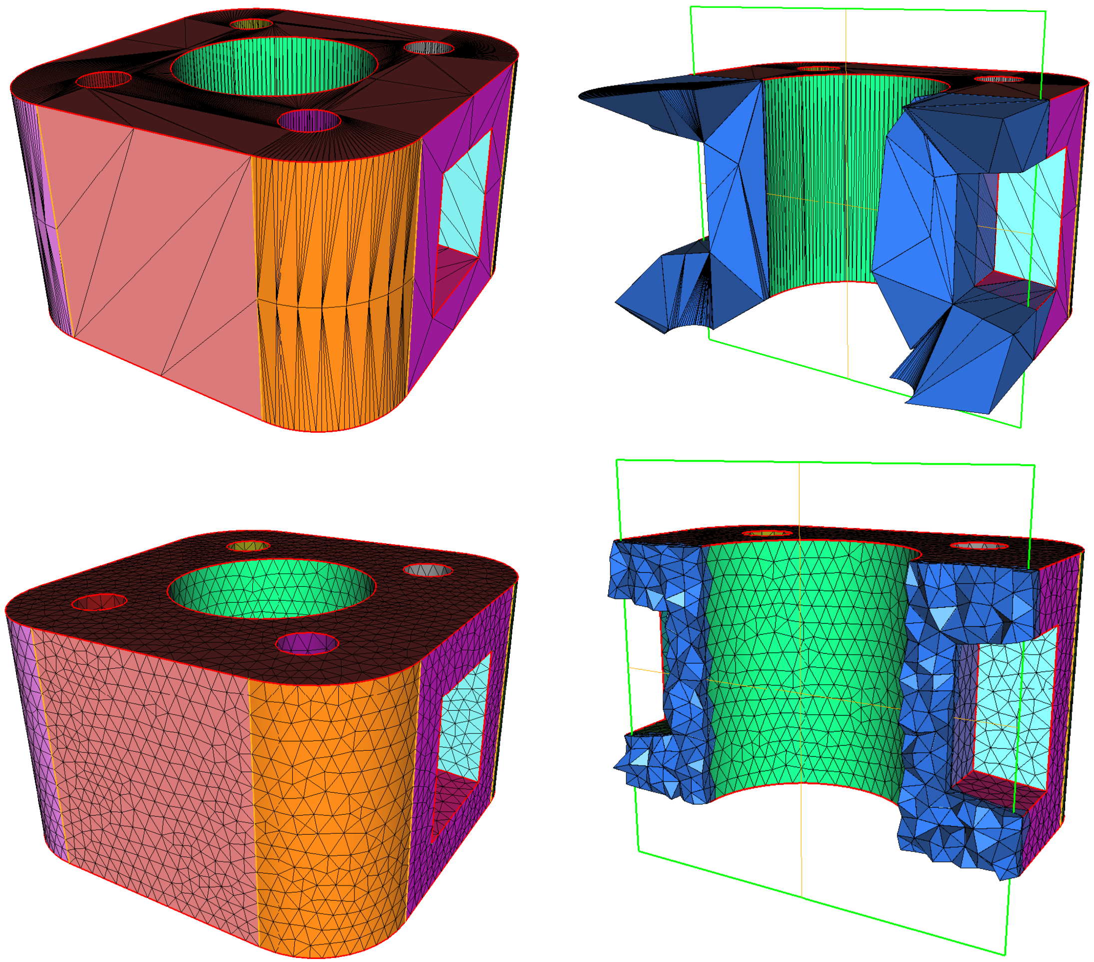


Figure 13: *Mechanical part before (left) and after (right) remeshing.*

Local remeshing in $3d$: numerical examples

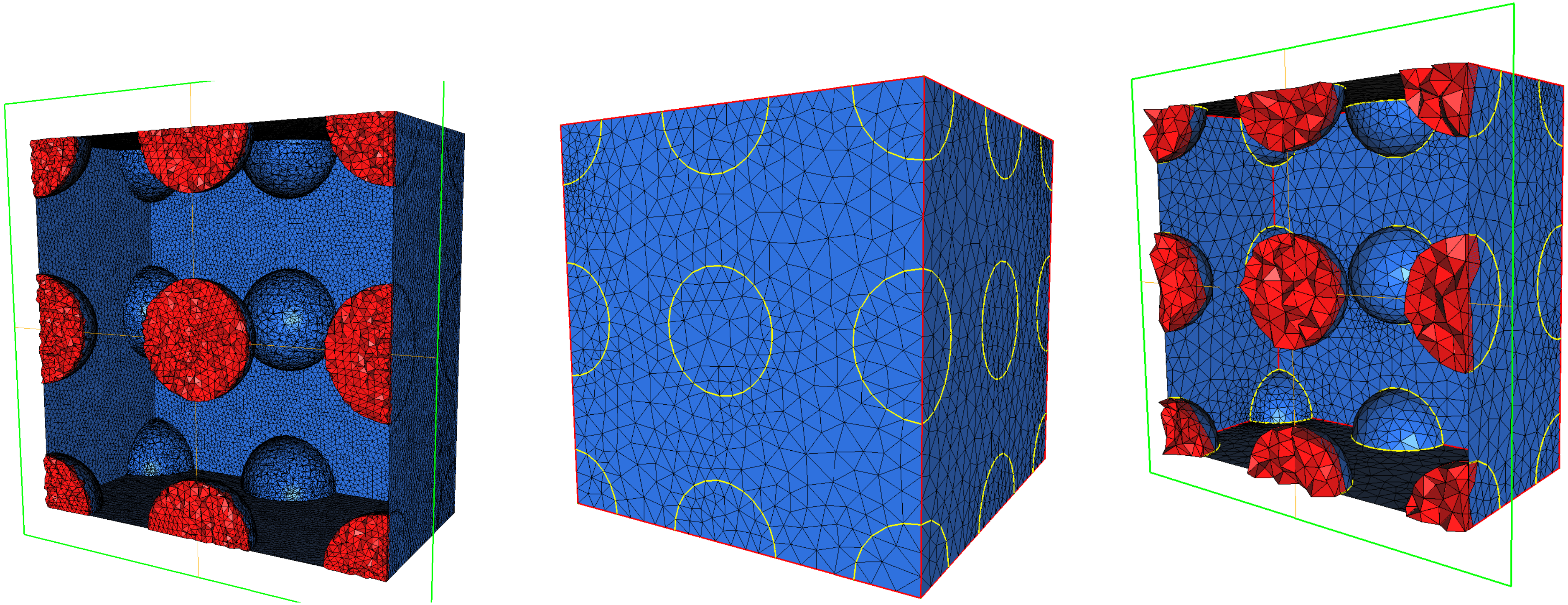


Figure 14: *(left) Ill-shaped discretization of an implicit function in a cube, (centre-right) result after local remeshing.*

Outline

I. Mathematical modeling of shape optimization problems

1. Differentiation with respect to the domain: Hadamard's method
2. Numerical implementation of shape optimization algorithms
3. The proposed method

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

1. A few words about the level set Method
2. Initializing level-set functions with the signed distance function
3. 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 D, in which a mesh of Ω^n explicitly appears as a submesh.

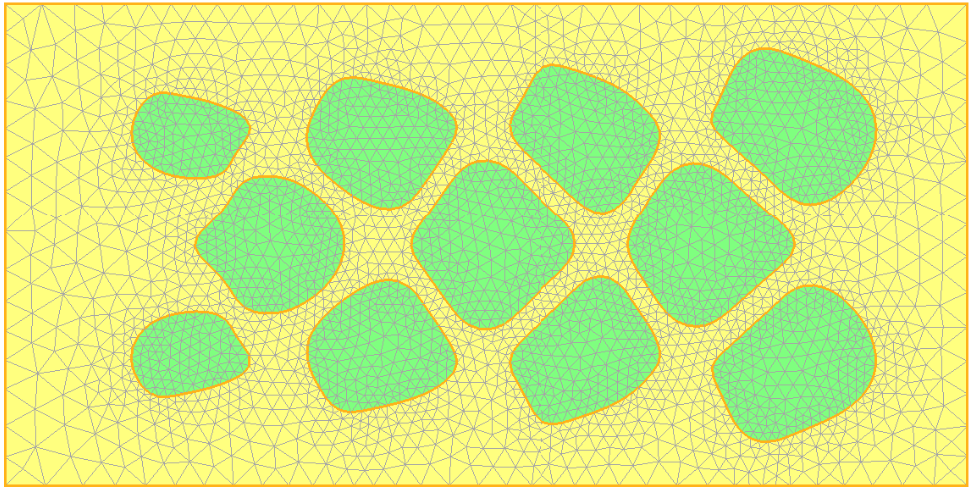
- When dealing with finite element computations on Ω^n, the part of $\mathcal{T}^n$, exterior to Ω^n is simply 'forgotten'.

- When dealing with the advection step, a level set function ϕ^n is generated on the whole mesh $\mathcal{T}^n$, and the level set advection equation is solved on this mesh, to get ϕ^{n+1}.

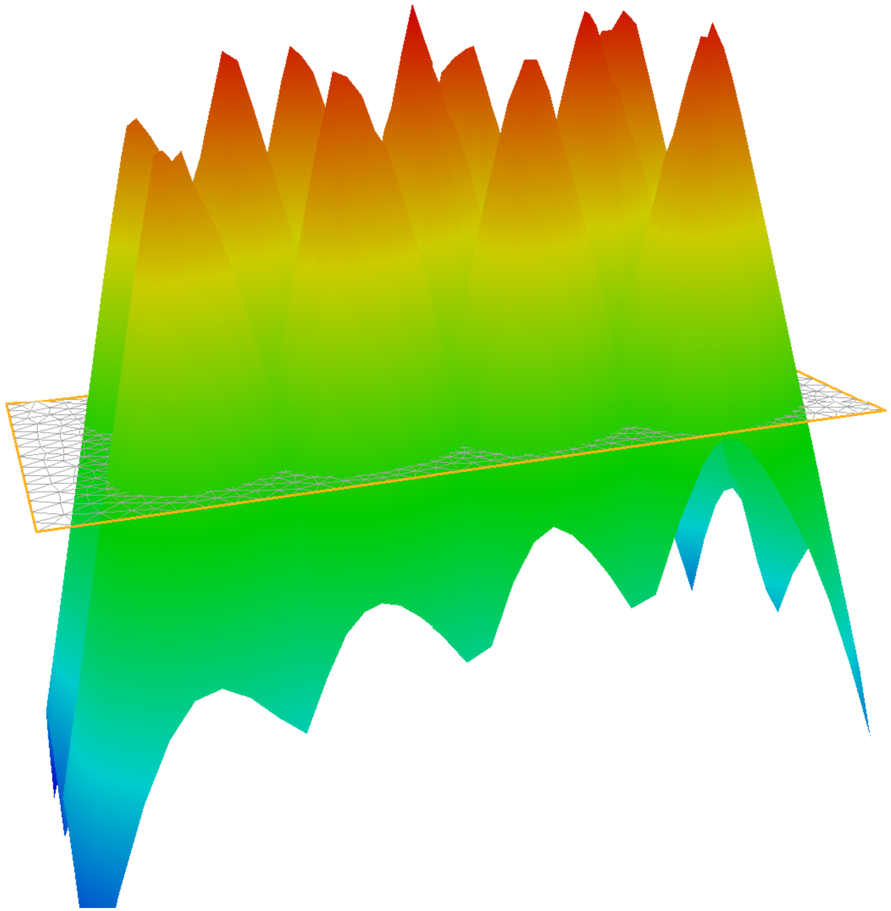
- From the knowledge of ϕ^{n+1}, a new unstructured mesh $\mathcal{T}^{n+1}$, in which the new shape Ω^{n+1} explicitly appears, is recovered.

The algorithm in motion...

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



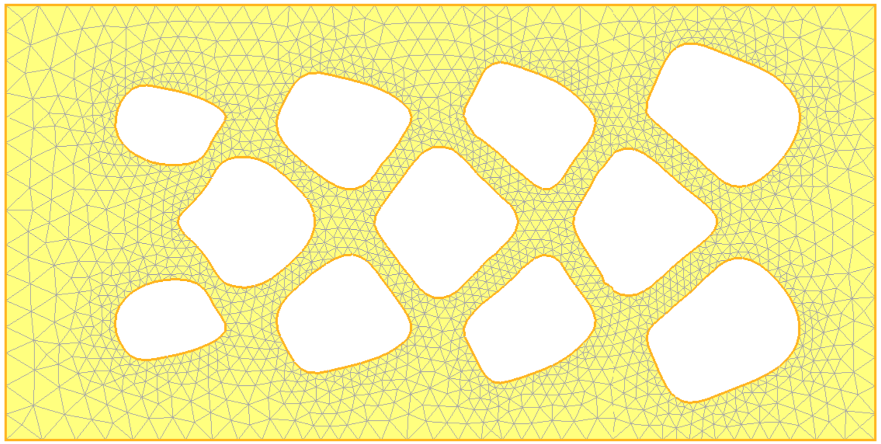
(a) The initial shape



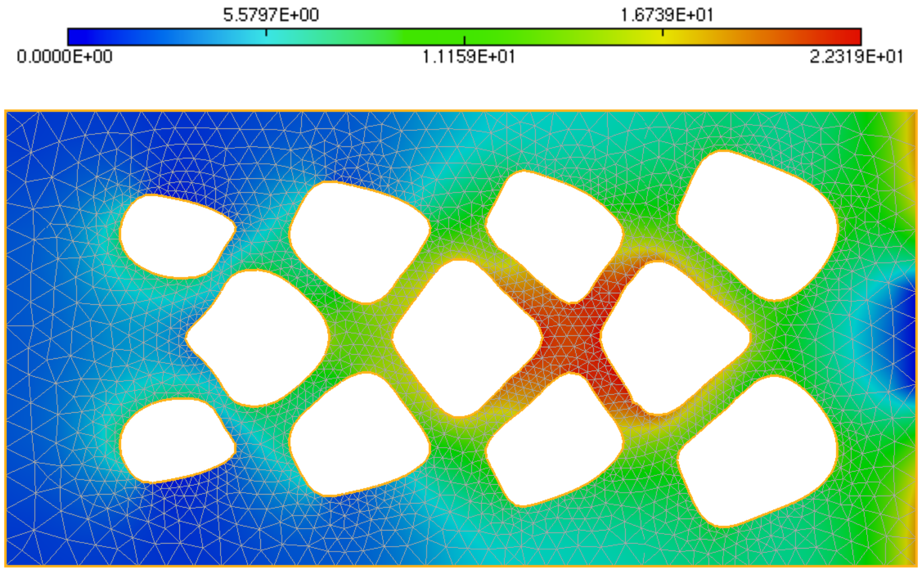
(b) Graph of d_{Ω^n}

The algorithm in motion...

Step 2: "Forget" the exterior of the shape $D \setminus \Omega^n$, and perform the computation of the shape gradient $J'(\Omega^n)$ on (the mesh of) Ω^n.



(a) The "interior mesh"



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

The algorithm in motion...

Step 3: "Remember" the whole mesh $\mathcal{T}^n$ of D. Extend the velocity field $J'(\Omega^n)$ to the whole mesh, and advect d_{Ω^n} along $J'(\Omega^n)$ 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}.

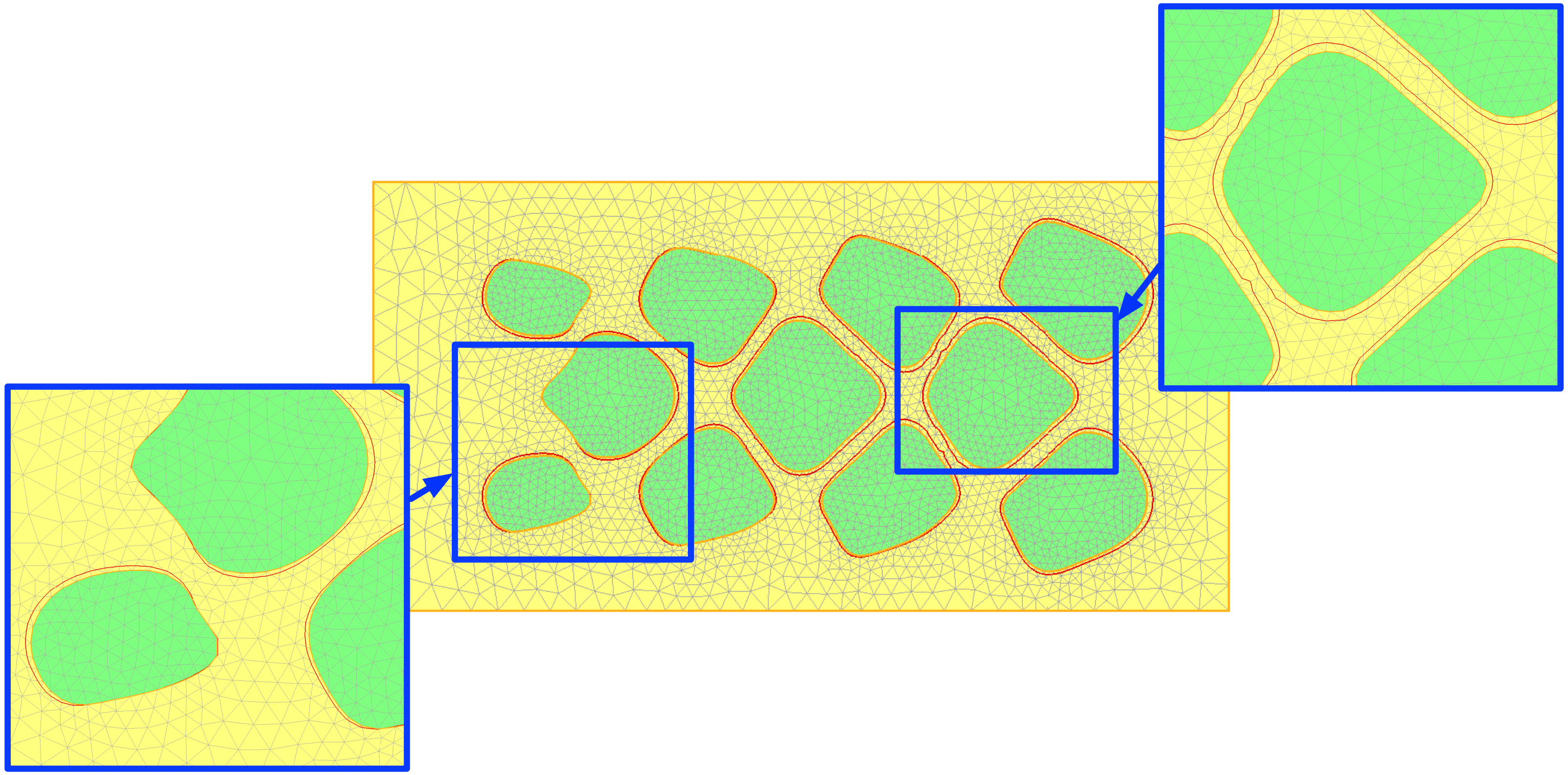


Figure 15: *The shape Ω^n, discretized in the mesh (in yellow), and the "new", advected 0-level set (in red).*

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.

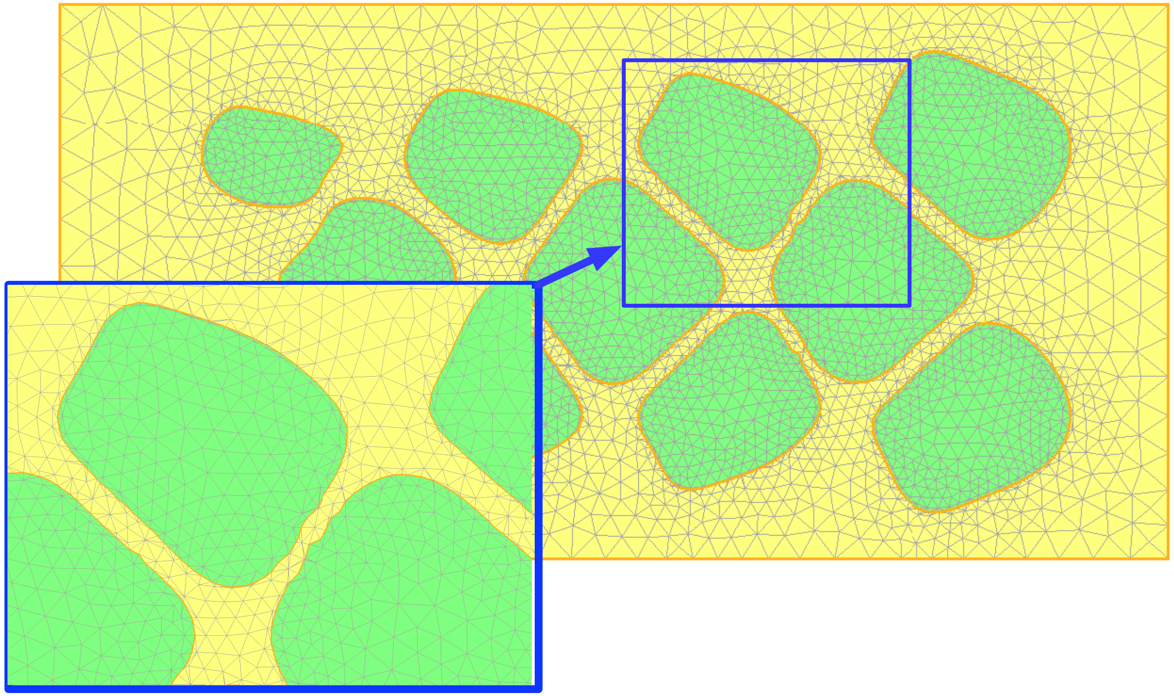


Figure 16: *Rough discretization of the* 0 *level set of* ϕ^{n+1} *into* $\mathcal{T}^n$*; the resulting mesh of* D *is ill-shaped.*

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.

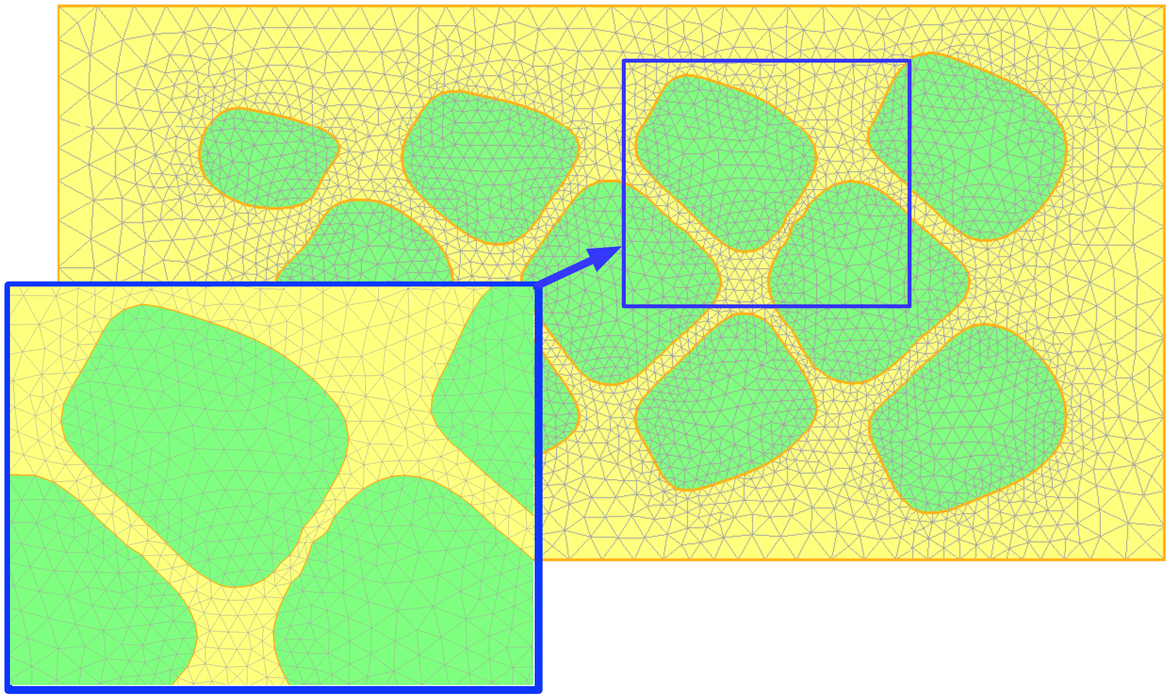
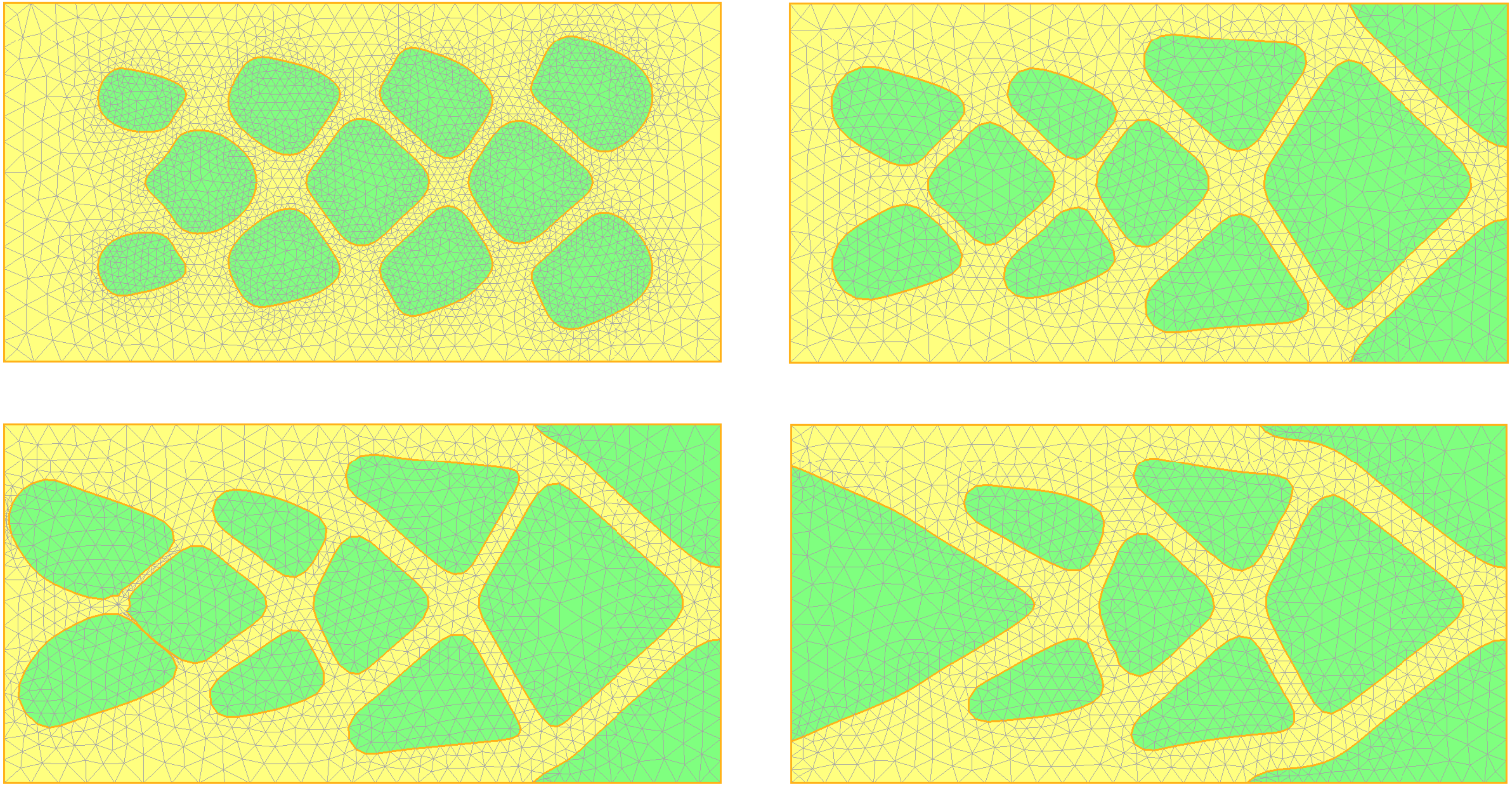


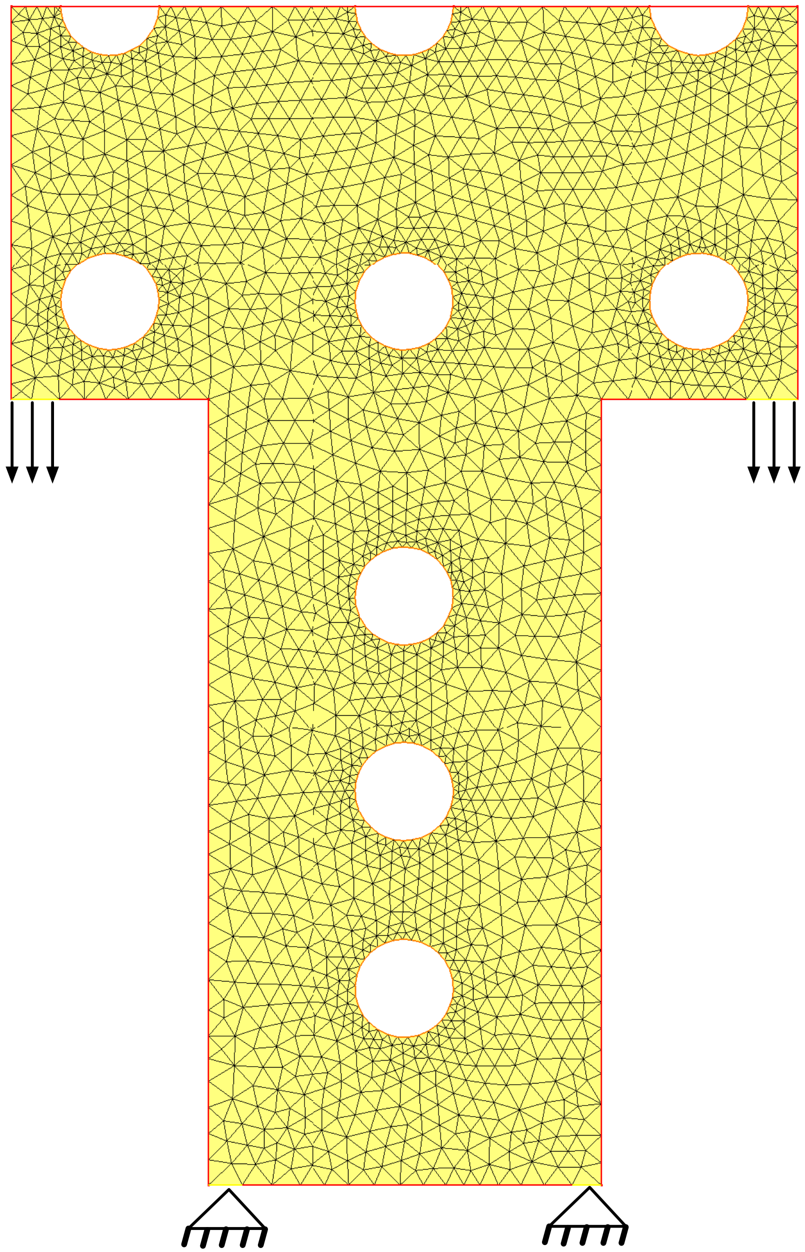
Figure 17: *Quality-oriented remeshing of the previous mesh ends with the new, well-shaped mesh* $\mathcal{T}^{n+1}$ *of* D *in which* Ω^{n+1} *is explicitly discretized.*

The algorithm in motion...

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



Numerical results: $2d$ optimal mast



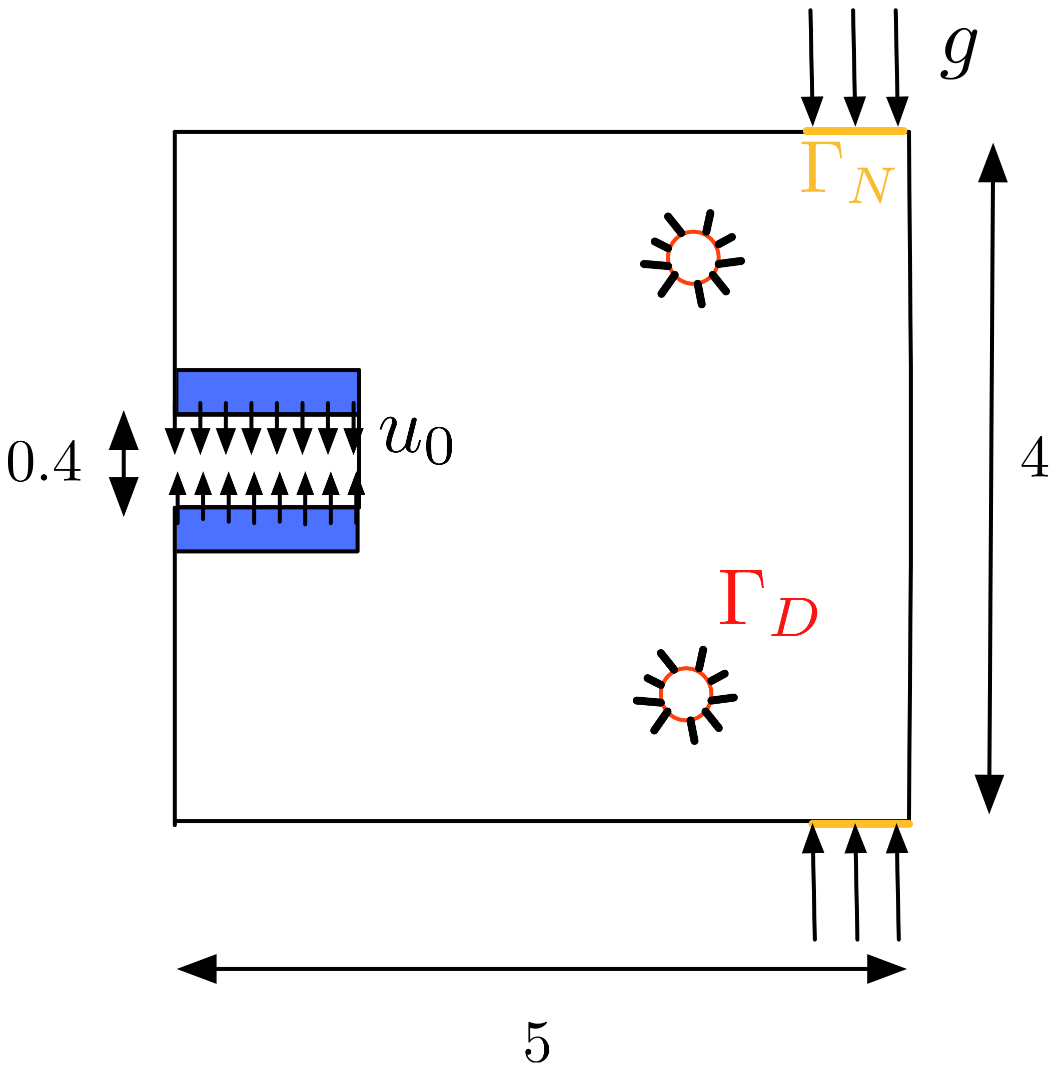
The 'benchmark' two-dimensional optimal mast test case.

- 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.

Numerical results: 2d gripping mechanism



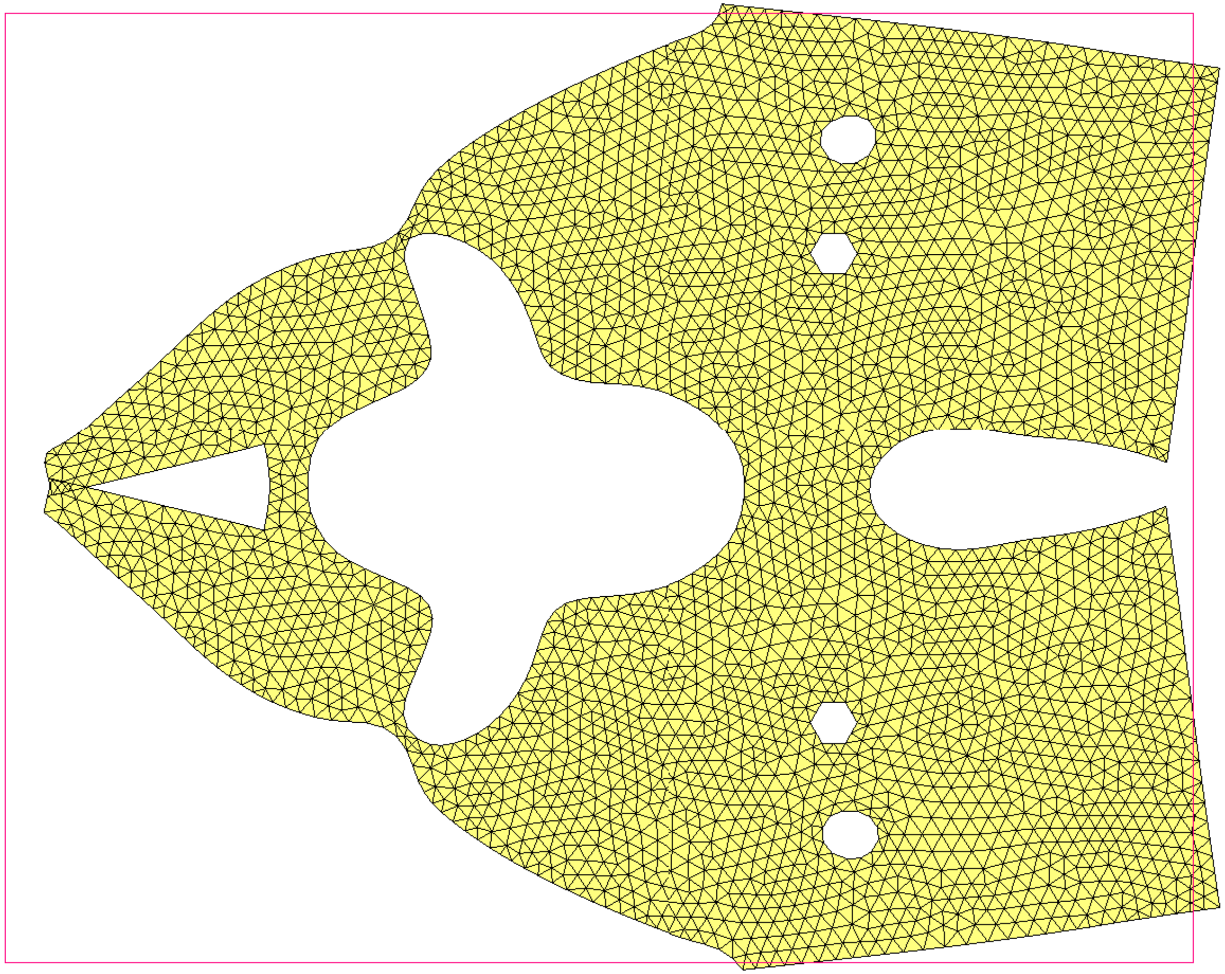
Device of a gripping mechanism.

The least-square criterion is minimized:

$$D(\Omega) = \int_\Omega k(x)||u_\Omega - u_0||^2\,dx,$$

where k is a the characteristic function of a region near the jaws, and u_0 is cooked so that the jaws close.

Numerical results: deformation of the optimal grip



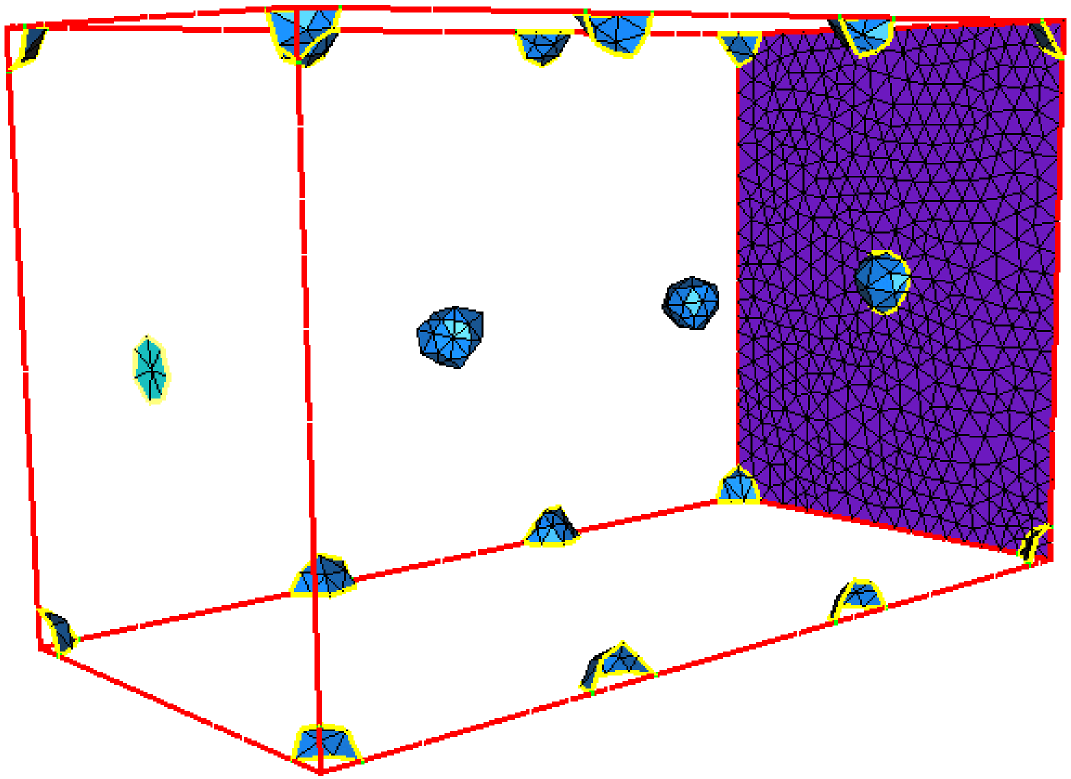
Numerical results: 3d cantilever

The 'benchmark' three-dimensional cantilever test case.

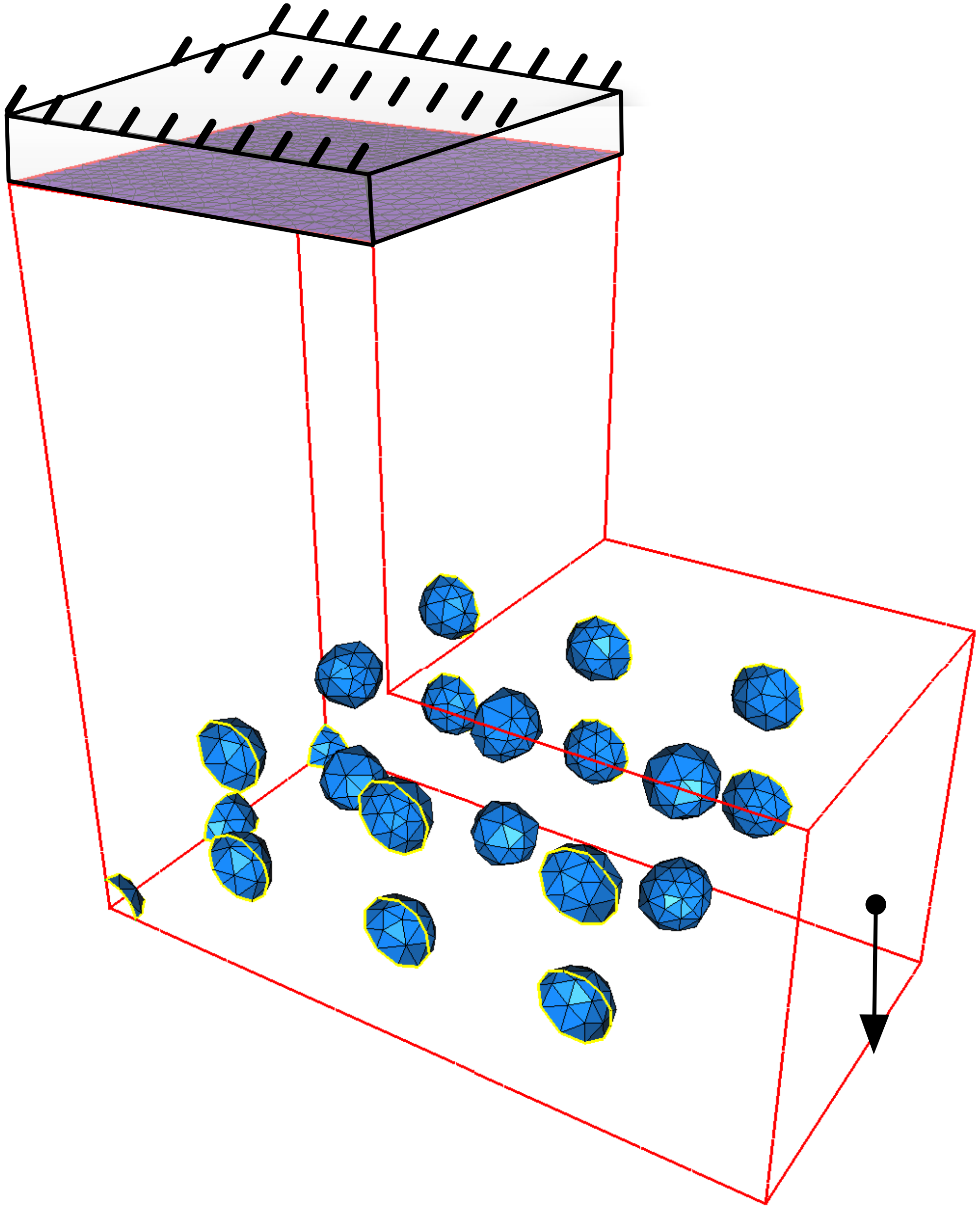
- Minimization of the compliance

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

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



Numerical results: $3d$ L-Beam



Optimal design of a $3d$ L-shaped beam.

- Minimization of a stress-based criterion

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

where k is a weight factor, and $\sigma(u) = Ae(u)$ is the stress tensor.

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

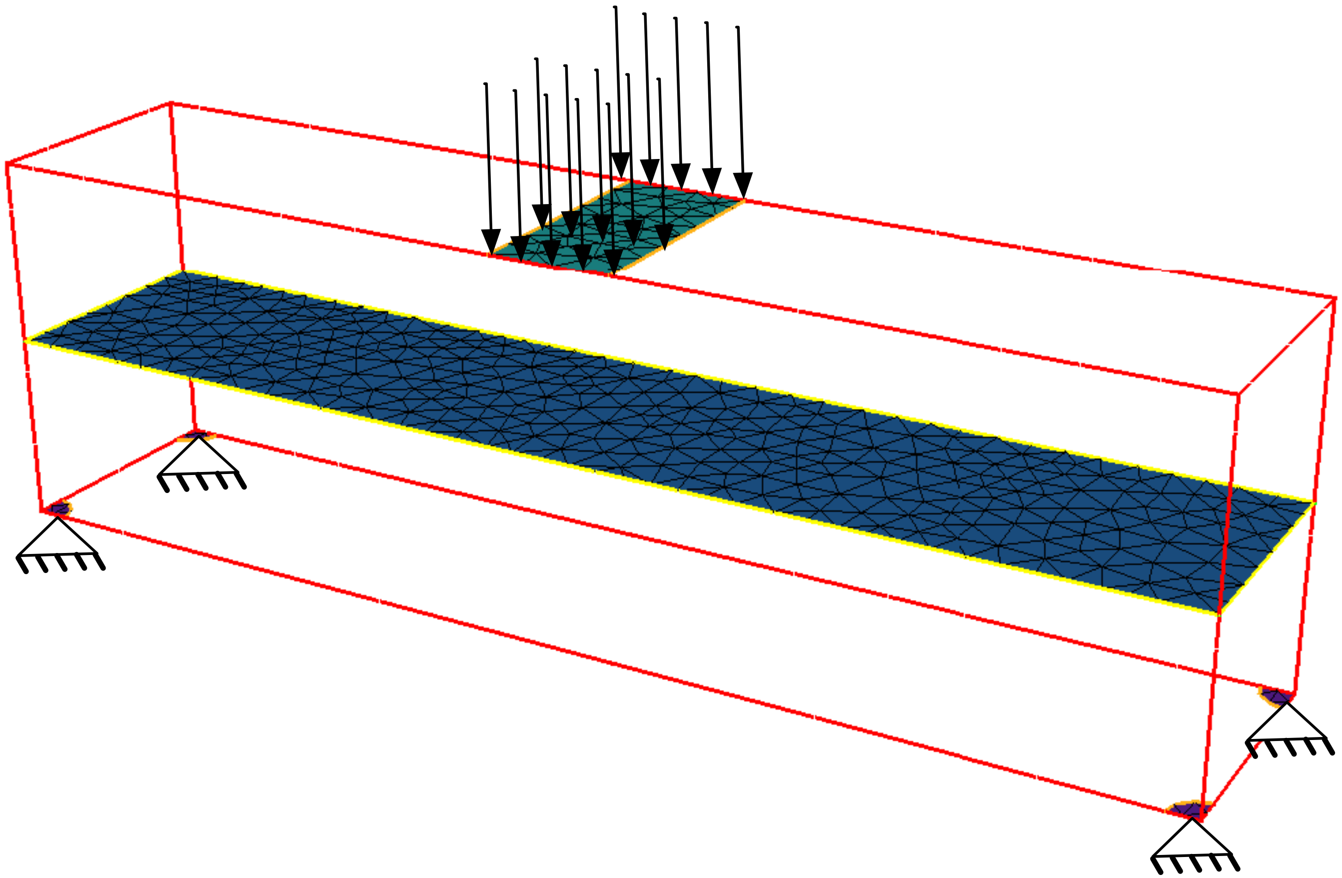
Numerical results: a multi-phase beam

Optimal repartition of two materials A_0, A_1 occupying subdomains Ω^0 and $\Omega^1 := D \setminus \Omega^0$ of a fixed beam D, with total (discontinuous) Hooke's law $A_{\Omega^0} := A_0\chi_{\Omega^0} + A_1\chi_{\Omega^1}$.

- Minimization of the total compliance of D:

$$C(\Omega^0) = \int_D A_{\Omega^0} e(u_{\Omega^0}) : e(u_{\Omega^0})\, dx.$$

- A constraint on the volume of the stronger material is enforced by means of a fixed Lagrange multiplier.



Thank you !