

# A level-set based mesh evolution method for shape optimization

Grégoire Allaire<sup>1</sup>, Charles Dapogny<sup>2</sup>, Florian Feppon<sup>1,3</sup>, Pascal Frey<sup>4</sup>

<sup>1</sup> CMAP, UMR 7641 École Polytechnique, Palaiseau, France

<sup>2</sup> Laboratoire Jean Kuntzmann, Université Joseph Fourier, Grenoble, France

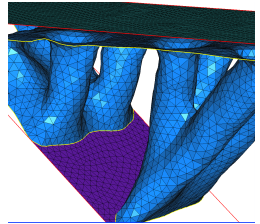
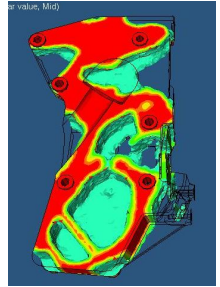
<sup>3</sup> Safran Tech, Magny-les-Hameaux, France

<sup>4</sup> Laboratoire J.-L. Lions, UPMC, Paris, France

3<sup>rd</sup> February, 2022

# Shape optimization and industrial applications

- The **increase in the cost of raw materials** urges to optimize the shape of mechanical parts from the early stages of design.
- The numerical resolution of **shape optimization problems** is plagued by a major difficulty:
  - The evaluations of the objective and its derivative involve **mechanical computations**, using the **Finite Element method** on a **mesh** of the shape.
  - The shape is (dramatically!) modified in the course of the iterative optimization process  
⇒ Need to **update this computational mesh**.
- This difficulty arises in many inverse problems: shape detection or reconstruction, image segmentation, etc.



- 1 Mathematical modeling of shape optimization problems
  - shape optimization of linear elastic structures
  - Differentiation with respect to the domain: Hadamard's method
  - Numerical implementation of shape optimization algorithms
  - The proposed method
  
- 2 From meshed domains to a level set description,... and conversely
  - Initializing level set functions with the signed distance function
  - Meshing the negative subdomain of a level set function: local remeshing
  
- 3 Application to shape optimization
  - Numerical implementation
  - The algorithm in motion
  - Numerical results

- 1 Mathematical modeling of shape optimization problems
  - shape optimization of linear elastic structures
    - Differentiation with respect to the domain: Hadamard's method
    - Numerical implementation of shape optimization algorithms
    - The proposed method
- 2 From meshed domains to a level set description,... and conversely
  - Initializing level set functions with the signed distance function
  - Meshing the negative subdomain of a level set function: local remeshing
- 3 Application to shape optimization
  - Numerical implementation
  - The algorithm in motion
  - Numerical results

## A model problem in linear elasticity

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

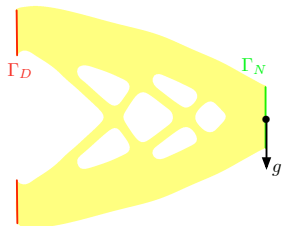
- **fixed** on a part  $\Gamma_D$  of its boundary,
- submitted to **surface loads**  $g$ , applied on  $\Gamma_N \subset \partial\Omega$ ,  $\Gamma_D \cap \Gamma_N = \emptyset$ .

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

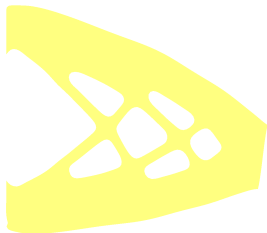
$$\begin{cases} -\operatorname{div}(Ae(u_\Omega)) &= 0 & \text{in } \Omega \\ u_\Omega &= 0 & \text{on } \Gamma_D \\ Ae(u_\Omega)n &= g & \text{on } \Gamma_N \\ Ae(u_\Omega)n &= 0 & \text{on } \Gamma \end{cases},$$

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

$$\forall e \in \mathcal{S}_d(\mathbb{R}), \quad Ae = 2\mu e + \lambda \operatorname{tr}(e)I.$$



A "Cantilever"



The deformed cantilever

## A model problem in linear elasticity

**Goal:** Starting from an initial structure  $\Omega_0$ , find a new one  $\Omega$  that minimizes a certain functional of the domain  $J(\Omega)$ .

### Examples:

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

$$C(\Omega) = \int_{\Omega} A e(u_{\Omega}) : e(u_{\Omega}) dx = \int_{\Gamma_N} g \cdot u_{\Omega} ds$$

- A **least-square error** between  $u_{\Omega}$  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  $\alpha$  is a fixed parameter, and  $k(x)$  is a weight factor.

A **volume constraint** may be enforced with a fixed penalty parameter  $\ell$ :

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

## 1 Mathematical modeling of shape optimization problems

- shape optimization of linear elastic structures
- **Differentiation with respect to the domain: Hadamard's method**
- Numerical implementation of shape optimization algorithms
- The proposed method

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

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

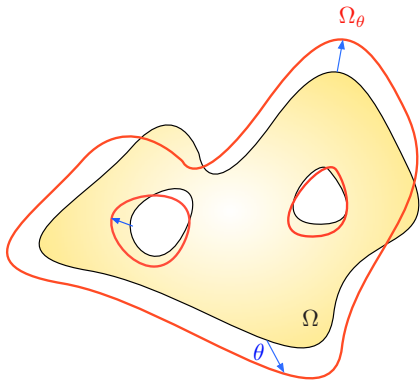
## 3 Application to shape optimization

- Numerical implementation
- The algorithm in motion
- Numerical results

**Hadamard's boundary variation method** features variations of a reference, Lipschitz domain  $\Omega$  of the form:

$$\Omega_\theta := (\text{Id} + \theta)(\Omega),$$

for “small”  $\theta \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ .



## Lemma 1.

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

### Definition 1.

Let  $\Omega \subset \mathbb{R}^d$  be a smooth domain. A (scalar) function  $\Omega \mapsto F(\Omega)$  is **shape differentiable** at  $\Omega$  if the mapping

$$W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \ni \theta \mapsto F(\Omega_\theta)$$

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

$$F(\Omega_\theta) = F(\Omega) + F'(\Omega)(\theta) + o(\|\theta\|_{W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)}).$$

The bounded operator  $\theta \mapsto F'(\Omega)(\theta)$  is the **shape derivative** of  $J(\Omega)$  at  $\Omega$ .

## Differentiation with respect to the domain: Hadamard's method (III)

- Techniques from optimal control allow to compute **shape derivatives**; in the case of “many” functionals  $J(\Omega)$ , the latter has the **structure**:

$$J'(\Omega)(\theta) = \int_{\Gamma} v_{\Omega} \theta \cdot n \, ds,$$

where  $v_{\Omega}$  is a scalar field depending on  $u_{\Omega}$ , and possibly on an **adjoint state**  $p_{\Omega}$ .

- The derivative  $J'(\Omega)(\theta)$  yields a natural **descent direction** for  $J(\Omega)$ : *for instance*, defining  $\theta$  as

$$\theta = -v_{\Omega} n$$

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

$$J(\Omega_{t\theta}) = J(\Omega) - t \int_{\Gamma} v_{\Omega}^2 \, ds + o(t) < J(\Omega)$$

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

- 1 Mathematical modeling of shape optimization problems
  - shape optimization of linear elastic structures
  - Differentiation with respect to the domain: Hadamard's method
  - Numerical implementation of shape optimization algorithms
  - The proposed method
  
- 2 From meshed domains to a level set description,... and conversely
  - Initializing level set functions with the signed distance function
  - Meshing the negative subdomain of a level set function: local remeshing
  
- 3 Application to shape optimization
  - Numerical implementation
  - The algorithm in motion
  - Numerical results

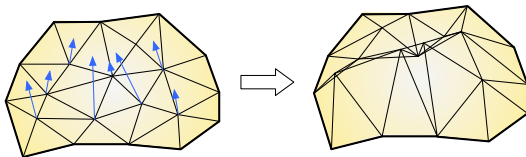
## The generic numerical algorithm

**Gradient algorithm:** For  $n = 0, \dots$  convergence,

- 1 Compute the solution  $u_{\Omega^n}$  (and  $p_{\Omega^n}$ ) of the elasticity system on  $\Omega^n$ .
- 2 From the **shape derivative**  $J'(\Omega^n)$ , infer a **descent direction**  $\theta^n$  for  $J(\Omega)$ .
- 3 **Advect** the shape  $\Omega^n$  according to  $\theta^n$ , so as to get  $\Omega^{n+1} := (\text{Id} + \theta^n)(\Omega^n)$ .

Problem: This strategy relies on two conflicting needs:

- An efficient **advection** of the shape  $\Omega^n \rightarrow \Omega^{n+1}$  at each step;
- A **high-quality mesh** of each shape  $\Omega^n$ , for finite element computations.



*Pushing nodes according to the velocity field may result in an invalid configuration.*

## The generic numerical algorithm (II)

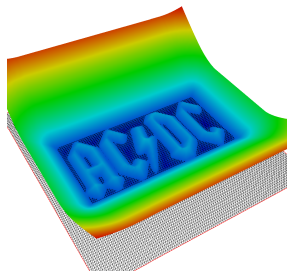
*Pushing the vertices of the mesh of  $\Omega^n$  along  $\theta^n$  inevitably makes it *ill-shaped*, or worse, *overlapping*.*

## A short detour by the Level Set Method

**A paradigm:** [OSe] *the motion of an evolving domain is best described in an **implicit** way.*

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

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



*A bounded domain  $\Omega \subset \mathbb{R}^2$  (left); graph of an associated level set function (right).*

## Surface evolution equations in the level set framework

- Let  $\Omega(t) \subset \mathbb{R}^d$  be a domain moving according to a velocity field  $v(t, x) \in \mathbb{R}^d$ .
- Let  $\phi(t, x)$  be a level set function for  $\Omega(t)$ .
- The motion of  $\Omega(t)$  translates in terms of  $\phi$  as the **level set advection equation**:

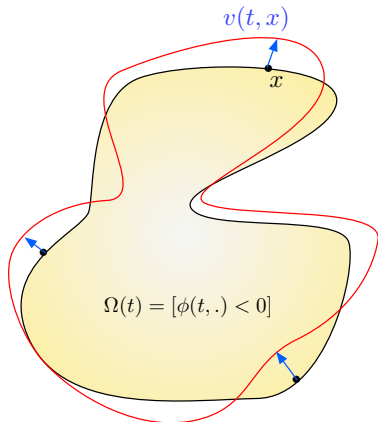
$$\frac{\partial \phi}{\partial t}(t, x) + v(t, x) \cdot \nabla \phi(t, x) = 0$$

- If  $v(t, x)$  is normal to the boundary  $\partial\Omega(t)$ , i.e.:

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

this rewrites as a **Hamilton-Jacobi equation**:

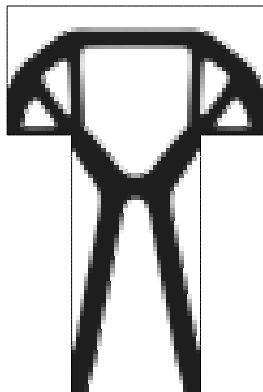
$$\frac{\partial \phi}{\partial t}(t, x) + V(t, x) |\nabla \phi(t, x)| = 0$$



$$\Omega(t + dt) = [\phi(t + dt, \cdot) < 0]$$

# The level set method of Allaire-Jouve-Toader [AlJouToa]

- The shapes  $\Omega^n$  are embedded in a working domain  $D$  equipped with a **fixed** mesh.
- The successive shapes  $\Omega^n$  are accounted for in the **level set** framework, i.e. via a function  $\phi^n : D \rightarrow \mathbb{R}$  which **implicitly** defines them.
- At each step  $n$ , the exact linear elasticity system on  $\Omega^n$  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** system posed on  $D$ .
- This approach is very versatile and does not require a mesh of the shapes at each iteration.



*Shape accounted for with a level set description*

- 1 Mathematical modeling of shape optimization problems
  - shape optimization of linear elastic structures
  - Differentiation with respect to the domain: Hadamard's method
  - Numerical implementation of shape optimization algorithms
  - The proposed method
  
- 2 From meshed domains to a level set description,... and conversely
  - Initializing level set functions with the signed distance function
  - Meshing the negative subdomain of a level set function: local remeshing
  
- 3 Application to shape optimization
  - Numerical implementation
  - The algorithm in motion
  - Numerical results

# The proposed method for handling mesh evolution

The mesh  $\mathcal{T}^n$  of  $D$  is **unstructured** and **changes at each iteration  $n$** , so that  $\Omega^n$  is **explicitly discretized in  $\mathcal{T}^n$** .

- Finite element analyses are conducted on  $\Omega^n$  by “forgetting” the part of  $\mathcal{T}^n$  for the void  $D \setminus \Omega^n$ .
- The advection  $\Omega^n \rightarrow \Omega^{n+1}$  is carried out on the whole mesh  $\mathcal{T}^n$ , using a level set description  $\phi^n$  of  $\Omega^n$ .

Computation of  
a descent direction  $\theta^n$

$$(\mathcal{T}^n, \Omega^n) \xrightarrow{\text{?}} (\mathcal{T}^{n+1}, \Omega^{n+1})$$

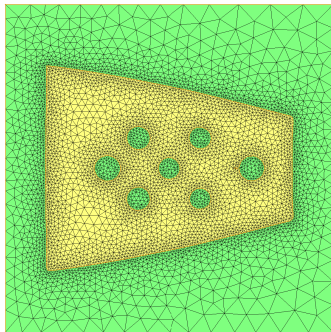
Generation of a  
level set function on  
an unstructured mesh

Explicit discretization of  
an implicit domain in  
the ambient mesh

$$(\mathcal{T}^n, \phi^n) \xrightarrow{\quad} (\mathcal{T}^n, \phi^{n+1})$$

Resolution of the advection  
equation on  $(0, \tau^n) \times D$ :

$$\begin{cases} \frac{\partial \phi}{\partial t} + \theta^n \cdot \nabla \phi = 0 \\ \phi(t = 0, \cdot) = \phi^n. \end{cases}$$



*Shape  $\Omega$  equipped with a mesh, conformingly embedded in a mesh of the box  $D$ .*

- 1 Mathematical modeling of shape optimization problems
  - shape optimization of linear elastic structures
  - Differentiation with respect to the domain: Hadamard's method
  - Numerical implementation of shape optimization algorithms
  - The proposed method
  
- 2 From meshed domains to a level set description,... and conversely
  - Initializing level set functions with the signed distance function
  - Meshing the negative subdomain of a level set function: local remeshing
  
- 3 Application to shape optimization
  - Numerical implementation
  - The algorithm in motion
  - Numerical results

## Isosurface discretization (II)

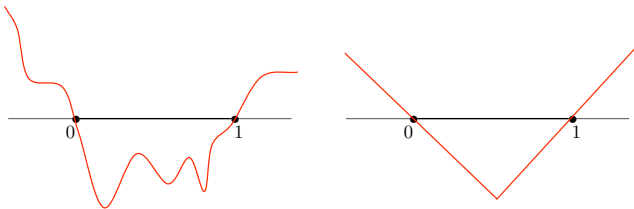
The level set function  $\phi$  for  $\Omega \subset D$  is often chosen as the **signed distance function**.

### Definition 2.

The **signed distance function**  $d_\Omega : \mathbb{R}^d \rightarrow \mathbb{R}$  to a bounded domain  $\Omega \subset \mathbb{R}^d$  is given by:

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

where  $d(x, \partial\Omega) := \min_{p \in \partial\Omega} |x - p|$  is the usual Euclidean distance from  $x$  to  $\partial\Omega$ .



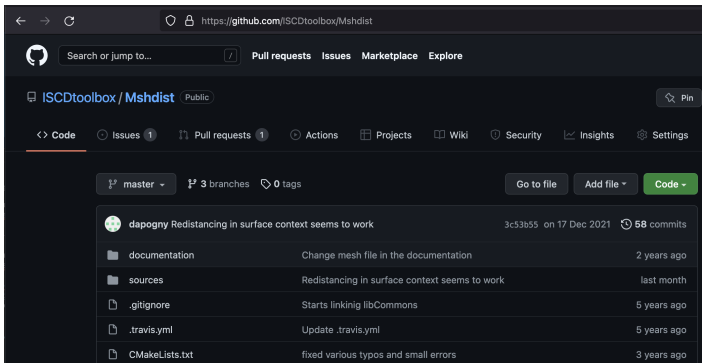
Two level set functions for the domain  $\Omega = (0, 1) \subset \mathbb{R}$ .

## Isosurface discretization (III)

- Efficient algorithms exist to calculate  $d_\Omega$ , such as the [Fast Marching algorithm](#) [SethianFMM], the [Fast Sweeping algorithm](#) [Zhao], etc.
- A free, open-source implementation: `mshdist` [DaFre].



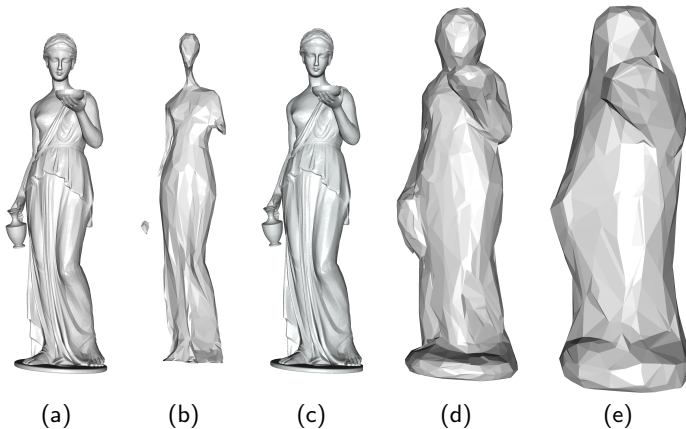
<https://github.com/ISCDtoolbox/Mshdist>



The screenshot shows the GitHub repository page for `ISCDtoolbox / Mshdist`. The repository is public and has 3 branches and 0 tags. The `Code` tab is selected, showing a list of files and their commit history:

File	Commit Message	Commit Date
<code>documentation</code>	Change mesh file in the documentation	2 years ago
<code>sources</code>	Redistancing in surface context seems to work	last month
<code>.gitignore</code>	Starts linking libCommons	5 years ago
<code>.travis.yml</code>	Update .travis.yml	5 years ago
<code>CMakeLists.txt</code>	fixed various typos and small errors	3 years ago

## A 3d example.

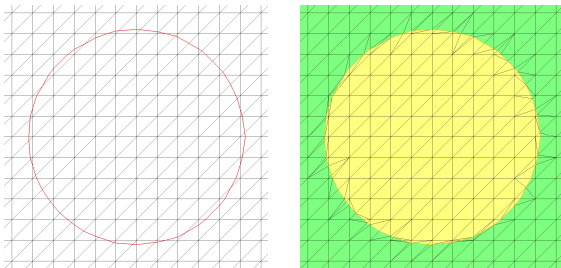


*Isosurfaces of the signed distance function to the 'Aphrodite' (a): (b): isosurface  $-0.01$ , (c): isosurface  $0$ , (d): isosurface  $0.02$ , (e): isosurface  $0.05$ .*

- 1 Mathematical modeling of shape optimization problems
  - shape optimization of linear elastic structures
  - Differentiation with respect to the domain: Hadamard's method
  - Numerical implementation of shape optimization algorithms
  - The proposed method
- 2 From meshed domains to a level set description,... and conversely
  - Initializing level set functions with the signed distance function
  - Meshing the negative subdomain of a level set function: local remeshing
- 3 Application to shape optimization
  - Numerical implementation
  - The algorithm in motion
  - Numerical results

## Meshing the negative subdomain of a level set function

Discretizing explicitly the 0 level set of a function  $\phi : D \rightarrow \mathbb{R}$  defined at the vertices of a simplicial mesh  $\mathcal{T}$  of a **computational box**  $D$  is fairly easy, using **patterns**.



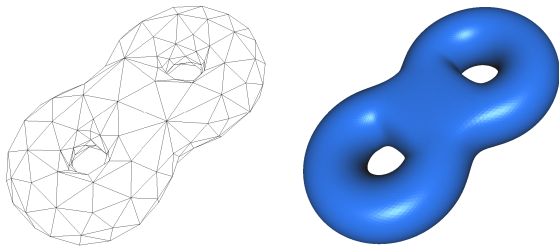
*(Left) 0 level set of a scalar function defined over a mesh; (right) explicit discretization in the mesh.*

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.

⇒ Need to improve the quality of a mesh, while retaining its geometric features.

## Local remeshing in 3d

- Let  $\mathcal{T}$  be an initial - valid, yet potentially ill-shaped - **tetrahedral mesh**.  $\mathcal{T}$  carries a **surface mesh**  $\mathcal{S}_{\mathcal{T}}$ , whose triangles are 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  $\partial\Omega$ .

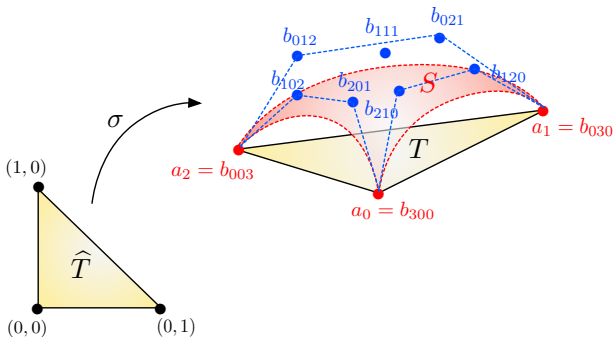


*Poor geometric approximation (left) of a domain with smooth boundary (right)*

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

## Local remeshing in 3d: definition of an ideal domain

- In realistic cases, the “ideal” domain  $\Omega$  of  $\mathcal{T}$  is unknown.
- However, from the knowledge of  $\mathcal{T}$  (and  $\mathcal{S}_{\mathcal{T}}$ ), one can **reconstruct geometric features of  $\Omega$  or  $\partial\Omega$** : normal vectors at regular points of  $\partial\Omega, \dots$
- These features allow for a **local parametrization** of  $\partial\Omega$  around each surface triangle  $T \in \mathcal{S}_{\mathcal{T}}$ , e.g. as a Bézier surface.



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

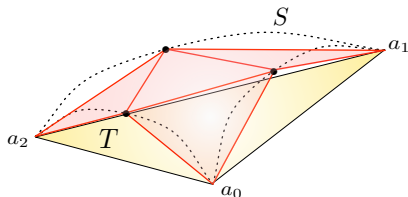
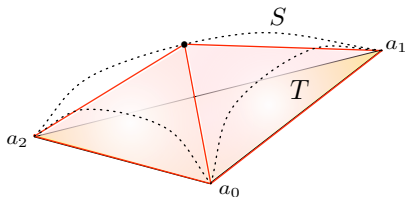
## Local remeshing in 3d: remeshing strategy

- 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  $\kappa_1, \kappa_2$  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  $T \in \mathcal{S}_T$ ,  $m$  lies on the piece of  $\partial\Omega$  computed from  $T$ . Else, it is merely inserted as the midpoint of  $p$  and  $q$ .
- An edge may be deemed “too long” when compared to the prescribed size, or because it entails a bad geometric approximation of  $\partial\Omega$ .

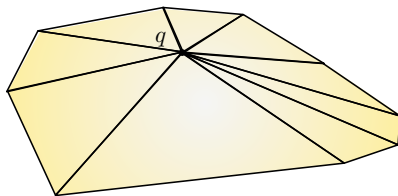
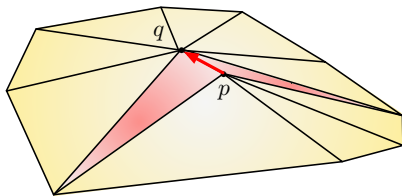


*Splitting of one (left) or three (right) edges of triangle  $T$ , positioning the new points on the ideal surface  $S$  (dotted).*

## Local mesh operators: edge collapse

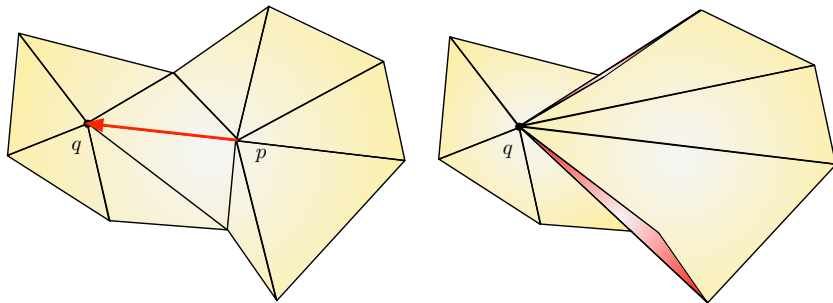
If an edge  $pq$  is “too short”, merge its two endpoints.

- Careful checks are in order to ensure the validity of the resulting configuration:
  - This operation may invalidate some tetrahedra (i.e. create overlappings).
  - When it is applied to a surface configuration, it may deteriorate the geometric approximation of  $\partial\Omega$ ;
- An edge may be “too short” when compared to the prescribed size, or because it is unnecessarily short for a fine geometric approximation of  $\partial\Omega$ .



*Collapse of point  $p$  over  $q$  in a surface configuration.*

## Local mesh operators: edge collapse

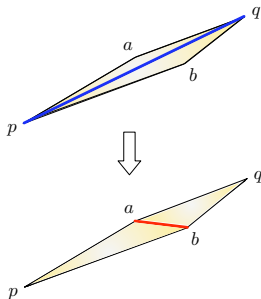


*In 2d, collapsing  $p$  over  $q$  (left) invalidates the resulting mesh (right): both greyed triangles end up inverted.*

## Local mesh operators: edge swap (I)

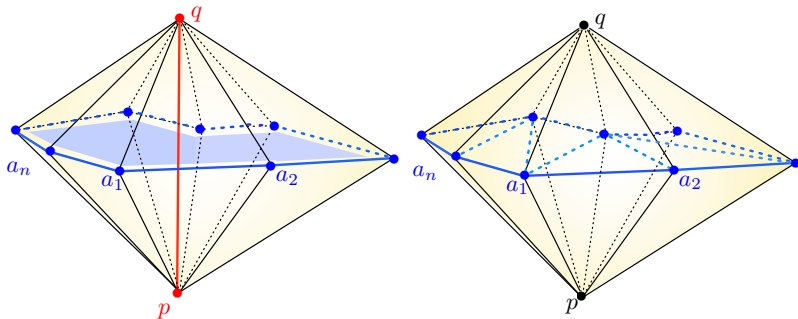
Suppress and edge  $pq$  from the mesh and reconnect the leftover cavity adequately.

This operator is key in improving the **quality** of the elements of the mesh.



*In 2d the edge  $pq$  is removed from the mesh, and the edge  $ab$  corresponding to the alternate configuration is added.*

## Local mesh operators: edge swap (II)

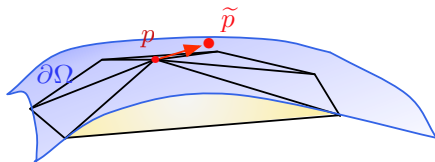


*The 3d edge swap operator is much more involved than its 2d counterpart.*

## Local mesh operators: node relocation

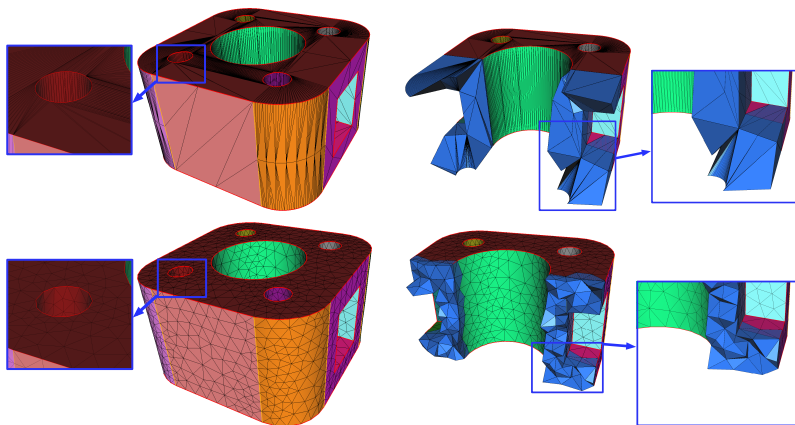
Slightly move a point  $p$  in the mesh, while leaving all connectivities unchanged.

This operator is the main ingredient in the fine-quality tuning of the mesh



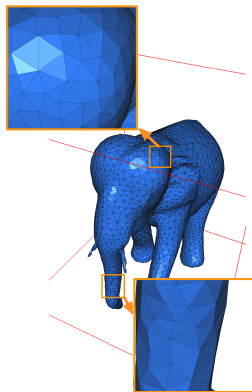
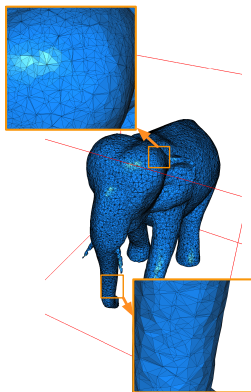
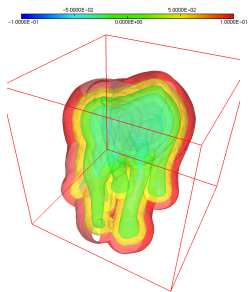
*Relocation of node  $p$  to  $\tilde{p}$ , along the surface.*

## Local remeshing in 3d: numerical examples



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

## Local remeshing in 3d: numerical examples



*(Left) Some isosurfaces of an implicit function defined in a cube, (centre) result after rough discretization in the ambient mesh, (right) result after local remeshing.*

## A word of advertisement

### Mmg PLATFORM

Robust, Open-source & Multidisciplinary  
Software for Remeshing



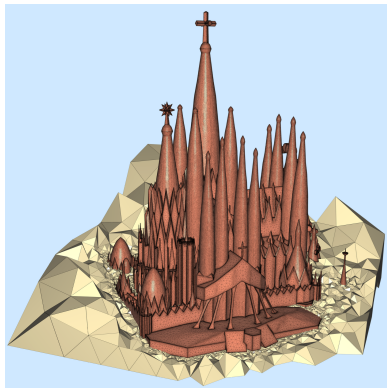
These general purpose remeshing algorithms are part of the **free, open-source** environment **Mmg**.



<https://www.mmgttools.org>



<https://github.com/MmgTools/mmg>



- 1 Mathematical modeling of shape optimization problems
  - shape optimization of linear elastic structures
  - Differentiation with respect to the domain: Hadamard's method
  - Numerical implementation of shape optimization algorithms
  - The proposed method
  
- 2 From meshed domains to a level set description,... and conversely
  - Initializing level set functions with the signed distance function
  - Meshing the negative subdomain of a level set function: local remeshing
  
- 3 Application to shape optimization
  - Numerical implementation
  - The algorithm in motion
  - Numerical results

## Numerical implementation

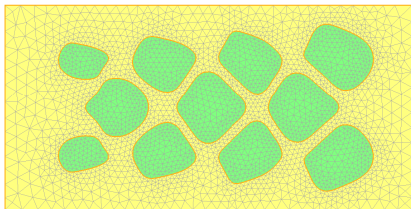
- At each iteration, the shape  $\Omega^n$  is endowed with an unstructured mesh  $\mathcal{T}^n$  of a larger, fixed, bounding box  $D$ ; a mesh of  $\Omega^n$  explicitly appears as a **submesh**.
- When dealing with **finite element computations** on  $\Omega^n$ , the part of  $\mathcal{T}^n$  exterior to  $\Omega^n$  is discarded.

⇒ The **shape gradient** is accurately calculated.

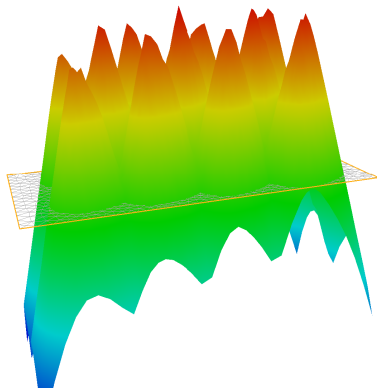
- When dealing with the **shape update** step,
  - ① A level set function  $\phi^n$  is generated on the **whole** mesh  $\mathcal{T}^n$ ,
  - ② The level set advection equation is solved on this mesh, to get  $\phi^{n+1}$ .
  - ③ From the knowledge of  $\phi^{n+1}$ , a new unstructured mesh  $\mathcal{T}^{n+1}$  is recovered, in which the new shape  $\Omega^{n+1}$  **explicitly appears**.

## The algorithm in motion...

**Step 1:** From the actual shape  $\Omega^n$ , generate the **signed distance function**  $d_{\Omega^n}$  at the vertices of the mesh  $\mathcal{T}^n$  of  $D$ .



(a) The initial shape

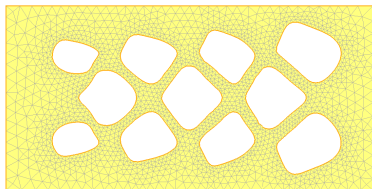


(b) Graph of  $d_{\Omega^n}$

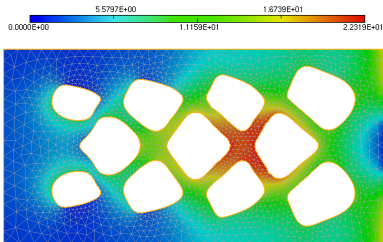
## The algorithm in motion...

### Step 2:

- Discard the exterior part  $D \setminus \overline{\Omega^n}$ ;
- Calculate the descent direction  $\theta^n$  on (the mesh  $\mathcal{T}^n$  of)  $\Omega^n$ .



(a) The "interior mesh"

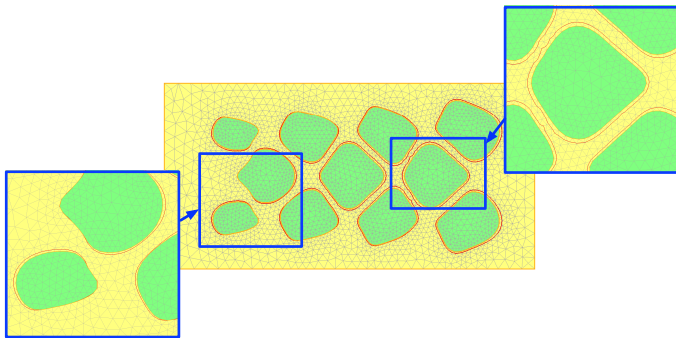


(b) Computation of  $\theta^n$

## The algorithm in motion...

### Step 3:

- "Retrieve" the whole mesh  $\mathcal{T}^n$  of  $D$ .
- Extend the velocity field  $\theta^n$  to the whole mesh;
- Advect  $d_{\Omega^n}$  along  $\theta^n$  for a (small) time step  $\tau^n$ .
- A new level set function  $\phi^{n+1}$  is obtained on  $\mathcal{T}^n$ , for the new shape  $\Omega^{n+1}$ .

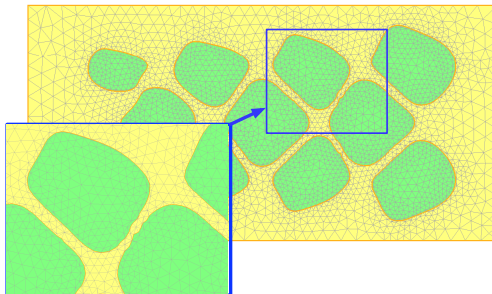


The shape  $\Omega^n$ , discretized in the mesh (in yellow), and the "new", advected 0-level set (in red).

## The algorithm in motion...

### Step 4:

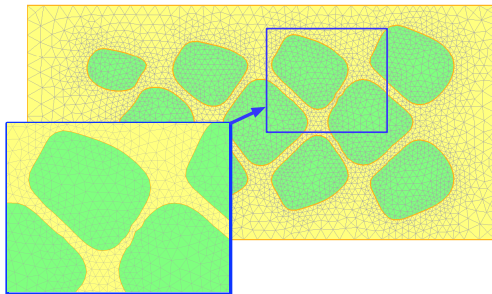
- The 0 level set of  $\phi^{n+1}$  is **explicitly discretized in the mesh  $\mathcal{T}^n$** .
- As expected, roughly "inserting" this line in  $\mathcal{T}^n$  yields a very ill-shaped mesh.



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

## The algorithm in motion...

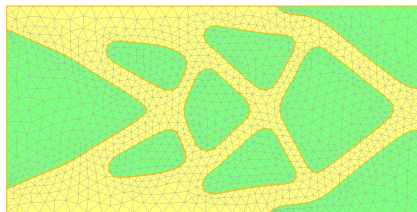
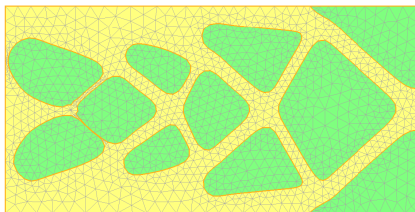
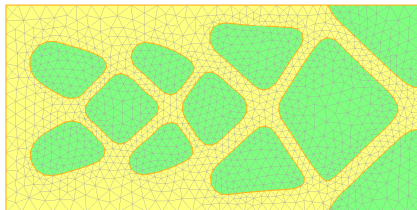
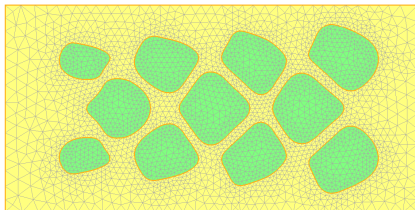
- **Mesh modification** is then conducted, so as to enhance the overall quality of the mesh according to the geometry of the shape.
- The new mesh  $\mathcal{T}^{n+1}$  is eventually obtained.



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

## The algorithm in motion...

Repeat the procedure 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**

$$C(\Omega) = \int_{\Omega} A e(u_{\Omega}) : e(u_{\Omega}) \, dx.$$

- A volume constraint is enforced.

## Numerical results: 3d cantilever

The “benchmark” three-dimensional cantilever test case.

- Minimization of the compliance

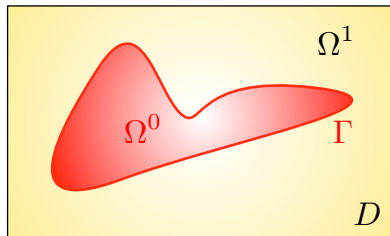
$$C(\Omega) = \int_{\Omega} A e(u_{\Omega}) : e(u_{\Omega}) \, dx.$$

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

## Another example in multiphase optimization

Optimal repartition of two materials  $A_0, A_1$  occupying subdomains  $\Omega^0$  and  $\Omega^1 := D \setminus \Omega^0$  of a fixed working domain  $D$ , with total (discontinuous) Hooke's law

$$A_{\Omega^0} := A_0 \chi_{\Omega^0} + A_1 \chi_{\Omega^1}.$$



- We minimize the compliance  $C(\Omega^0) = \int_D A_{\Omega^0} e(u_{\Omega^0}) : e(u_{\Omega^0}) \, dx$  of  $D$ .
- The shape derivative reads:

$$C'(\Omega^0)(\theta) = \int_{\Gamma} \mathcal{D}(u, u) \theta \cdot n \, ds.$$

- Evaluating  $\mathcal{D}(u, u)$  is awkward in a fixed mesh context, for it involves jumps of the (discontinuous) strain and stress tensors  $e(u)$  and  $\sigma(u)$  at the interface  $\Gamma$ .

## Numerical results: a multiphase beam

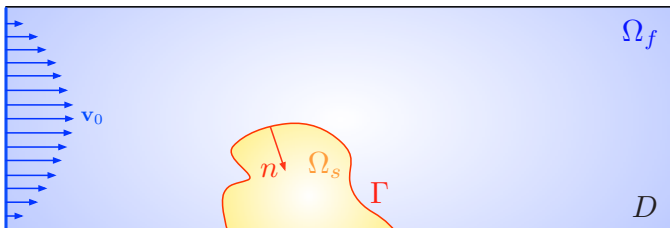
- We minimize the **compliance** of a beam  $D$ , with respect to the repartition of the constituent materials  $A_0, A_1$  ( $E^1 = E^0/3$ ).
- A constraint on the volume of the stiffer material is enforced by means of a fixed Lagrange multiplier.

## An advanced example in fluid-structure interaction (I)

- A solid obstacle  $\Omega_s := \Omega$  is placed inside a fixed cavity  $D$  where a fluid is flowing, occupying the phase  $\Omega_f := D \setminus \overline{\Omega_s}$ .
- The fluid obeys the **Navier-Stokes equations** ( $Re = 60$ ), and the solid is governed by the **linearized elasticity system**.
- **Weak coupling** between  $\Omega_f$  and  $\Omega_s$ : the fluid exerts a traction on the interface  $\Gamma$ .
- We optimize the shape of  $\Omega_s$  with respect to the **solid compliance**

$$J(\Omega) = \int_{\Omega_s} Ae(u_{\Omega_s}) : e(u_{\Omega_s}) \, dx,$$

under a volume constraint.



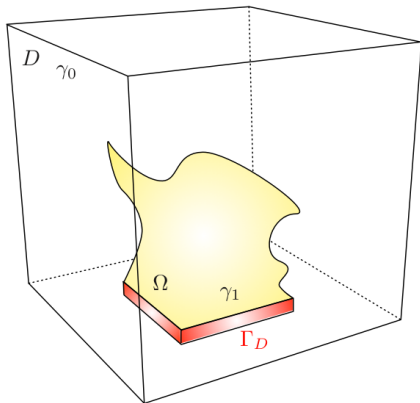
## An advanced example in fluid-structure interaction (II)

## Optimization of the shape of a heat diffuser (I)

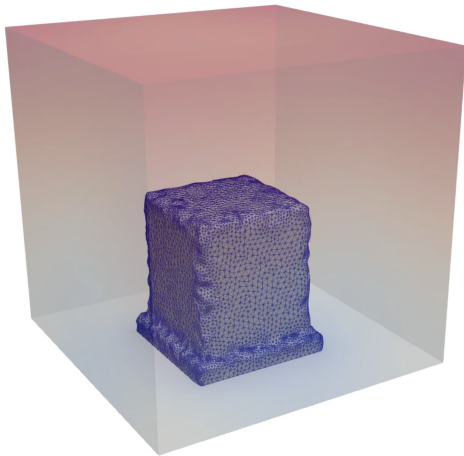
- A thermal chamber  $D$  is divided into
  - A phase  $\Omega$  with **high conductivity**  $\gamma_1$
  - A phase  $D \setminus \overline{\Omega}$  with **low conductivity**  $\gamma_0$ .
- A temperature  $T_0 = 0$  is imposed on  $\Gamma_D$  and the remaining boundary  $\partial D \setminus \overline{\Gamma_D}$  is insulated from the outside.
- A heat source is acting inside  $D$ .
- The temperature  $u_\Omega$  inside  $D$  is solution to the **two-phase** Laplace equation.
- The **average temperature** inside  $D$ ,

$$J(\Omega) = \frac{1}{|D|} \int_D u_\Omega \, dx$$

is minimized under a volume constraint.



## Optimization of the shape of a heat diffuser (II)



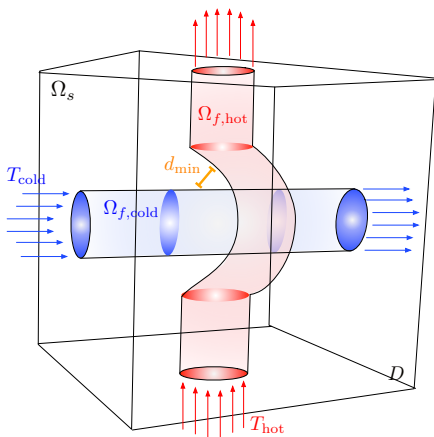
*Optimization of the shape of a heat diffuser.*

## Optimization of the shape of a heat exchanger (I)

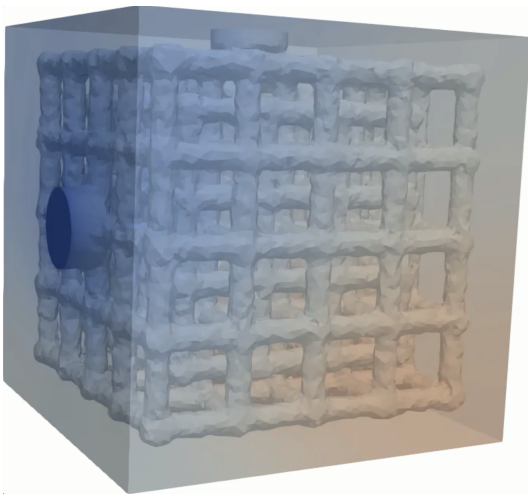
- A thermal chamber  $D$  is divided into
  - A phase  $\Omega_{f,hot}$  conveying a **hot fluid**;
  - A phase  $\Omega_{f,cold}$  conveying a **cold fluid**;
  - A **solid phase**  $\Omega_s$ .
- The **Navier-Stokes equations** are satisfied in  $\Omega_{f,hot}$ ,  $\Omega_{f,cold}$ .
- The **stationary heat equation** accounts for the temperature diffusion within  $D$ .
- The **heat transferred** from  $\Omega_{f,hot}$  to  $\Omega_{f,cold}$  is maximized.
- A constraint is imposed on the **minimal distance** between  $\Omega_{f,hot}$  and  $\Omega_{f,cold}$ :

$$d(\Omega_{f,hot}, \Omega_{f,cold}) \geq d_{min}.$$

- Volume and **pressure drop** constraints are added on  $\Omega_{f,hot}$ ,  $\Omega_{f,cold}$ .



## Optimization of the shape of a heat exchanger (II)



*Optimization of the shape of a heat exchanger.*

Thank you !

## References I



[AdDaFre] G. Allaire, C. Dapogny, and P. Frey, *Shape optimization with a level set based mesh evolution method*, Comput Methods Appl Mech Eng, 282 (2014), pp. 22–53.



[AlJouToa] G. Allaire, F. Jouve and A.-M. Toader, *Structural optimization using sensitivity analysis and a level-set method*, Journal of computational physics, 194 (2004), pp. 363–393.



[AuAu] J.F. Aujol and G. Aubert, *Signed distance functions and viscosity solutions of discontinuous Hamilton-Jacobi Equations*, INRIA Technical Report, 4507 (2002).



[Cho] D. Chopp, *Computing minimal surfaces via level-set curvature flow*, J. Comput. Phys., 106, pp. 77–91 (1993).



[DaFre] C. Dapogny and P. Frey, *Computation of the signed distance function to a discrete contour on adapted triangulation*, Calcolo, 49(3), (2012), pp. 193–219.

## References II



[DaDobFre] C. Dapogny, C. Dobrzinsky and P. Frey, *Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems*, J. Comput. Phys., 262, (2014), pp. 358–378.



[FepAlBorCorDa] F. Feppon, G. Allaire, F. Bordeu, J. Cortial and C. Dapogny, *Shape optimization of a coupled thermal fluid-structure problem in a level set mesh evolution framework*, SeMA J., 76, 3, (2019), pp. 413–458.



[FepAlDa] F. Feppon, G. Allaire and C. Dapogny, *Topology optimization of thermal fluid–structure systems using body-fitted meshes and parallel computing*, J. Comput. Phys., 417, 109574, (2020).



[FreGeo] P.J. Frey and P.L. George, *Mesh Generation : Application to Finite Elements*, Wiley, 2nd Edition, (2008).



[MuSi] F. Murat and J. Simon, *Sur le contrôle par un domaine géométrique*, Technical Report RR-76015, Laboratoire d'Analyse Numérique (1976).

## References III



[OSe] S.J. Osher and J.A. Sethian, *Fronts propagating with curvature-dependent speed : Algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.



[SethianFMM] J.A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci. USA Vol. 93, (1996), pp. 1591–1595.



[Zhao] H. Zhao, *A fast sweeping method for eikonal equations*, Mathematics of computation, 74(250), (2005), pp. 603–627.