Anisotropic adaptive body-fitted meshes for CFD

Sacha El Aouad, Aurélien Larcher, Elie Hachem

Anisotropic adaptive body-fitted meshes for CFD

Sacha El Aouad, Aurélien Larcher, Elie Hachem*

MINES ParisTech, PSL - Research University, CEMEF - Centre for Material Forming, CNRS UMR 7635,CS 10207 rue Claude Daunesse, 06904 Sophia-Antipolis Cedex, France

Abstract Mesh adaptation for immersed solid is one of the most challenging topics in computational mechanics. In this contribution, we propose a novel anisotropic adaptive body-fitted mesh method in particular for Computational Fluid Dynamics (CFD) where the boundary layers and the high gradient of a solution at a fluid-solid interface play a major role. It is characterized by its simplicity to implement, and by its generality to tackle complex geometries without additional efforts. It combines two successive iterations, knowing that the order of each one is important. First, a gradient based metric construction uses the gradients of the level-set of any immersed object to generate an anisotropic well adapted mesh. Since the elements are well stretched along the interface, and therefore the interface passes through these elements, a second step is implemented and applied. It is based on R-adaptation and swapping, to provide a sharp anisotropic fitted mesh. The proposed test cases, with and without a CFD solution, show the flexibility and the accuracy of the proposed automated h- and r- adaption framework designed for immersed methods.

Keywords Body-fitted mesh · Anisotropic adaptation · Immersed methods · Level-set function · R-adaptation · Swapping · FSI · CFD

1 Introduction

The development of efficient methods to simulate multi-components systems is among engineering challenges and still a need for industrials, especially in the case of fluid-structure interaction or conjugate heat transfer. In recent years, there has been an increasing interest in studying numerically a variety of engineering applications that involve such couplings between fluid and solid domains [38, 40, 41].

Classically, coupling techniques consist of dividing the global domain into several local subdomains over each of which a local model (equation to be solved) can be analyzed independently. The global solution can then be constructed by suitably piecing together local solutions from individually modeled subdomains [1, 10]. A body-fitted mesh is generally required for such simulations and its construction may be limited due either to the needed computational costs or to the complexity of the treated geometries. Alternatively, the development of immersed or embedded methods is becoming a subject of intense research mainly because of their relative simplicity and computational efficiency for simulating complex geometries [8, 9, 29]. Indeed, there is no constraint for the mesh generation, different approaches are proposed to enforce strongly or weakly the boundary conditions at the fluid-solid interface [2, 35, 34, 5, 6], and finally several clever techniques such as penalty or enrichment methods can be used to ensure continuity and to increase accuracy at the interface [4, 3, 7, 28, 38, 30, 37]. One common main challenge related to the development of these methods is in fact that elements at the interface are cut in such a way that a fraction of them remains inside the solid domain, and the other in the fluid domain, and consequently the question how to impose a boundary condition at the interface, or the search for a penalty approach, or the need for interface enrichment become justified.

*Correponding author: MINES ParisTech, PSL - Research University, CEMEF - Centre for Material Forming, CNRS UMR 7635,CS 10207 rue Claude Daunesse, 06904 Sophia-Antipolis Cedex, France
elie.hachem@mines-paristech.fr (E.Hachem) aurelien.larcher@mines-paristech.fr (A.Larcher) sacha.el_aouad@mines-paristech.fr (S.El Aouad)

Authors of [5, 6] working on the Shifted Boundary Method (SBM) describe very well this issue and propose to shift numerically the boundary conditions to fit the interface to preserve optimal convergence rates of the numerical solution. The method was it successfully applied to a Poisson and a Stokes problems. The SBM modifies the governing equations of the problem by weakly imposing the Dirichlet boundary conditions on a surrogate boundary using a Taylor expansion formulation. Using Nitsche's method, a penalty term is introduced to the equations to be solved.

The Ghost Cell Method [4], introduce the forcing term into the governing equations after discretization; the forcing term is extracted directly from the numerical solution. The governing equations are discretized on a computational grid, resulting in a set of discretized equations. The forcing term is then applied for cell points close to the immersed boundary to account for it.

However, this same challenge becomes even more intense when dealing with not only complex geometries, but also complex physical simulations or real-life simulations when for instance boundary layers are needed for turbulent flows. In this case, the need and the use of anisotropic mesh adaptation become desirable and justified.

In this work, we propose a new approach that leverages interface resolution/accuracy strategy and tackles both challenges: first, the desired local geometry resolution of a body-fitted mesh, thus eliminating the issue of a cut element, and second the needed numerical accuracy at the interface obtained by anisotropic unstructured mesh accounting for real-life practical applications.

Indeed, the proposed anisotropic adaptive body-fitted mesh approach is characterized first by its simplicity to implement, and by its robustness and generality to tackle complex geometries and physics without additional efforts. The algorithm is automatic, hence, it doesn't require any interference from the user and can start from any initial mesh. It is also adaptive evolving dynamically with both the interface and the computed solution fields. And finally, the approach combines both the advantages of anisotropic mesh, with its flexibility to capture the curvature of any complex geometry and is needed for high gradients and complex physics such as high Reynolds number, as well as the body-fitted mesh giving a sharp interface.

The anisotropic adaptive body-fitted combines two successive iterations knowing that the order of each step is important. First, a gradient based metric construction uses the gradients of the levelset of any immersed object to generate an anisotropic well adapted mesh [32, 33, 31, 14, 27]. Since the elements are well stretched along the interface, and therefore the interface passes through these elements, a second step is implemented and applied. It is based on R-adaptation and swapping, to provide a sharp anisotropic fitted mesh. Knowing also that the metric map can also be constructed using the gradient of any field such as the velocity, temperature, etc. [10, 11], we can then solve and capture the solution more accurately in the entire domain as well as on the immersed solid with a sharp and precise interface. Therefore, we propose several test cases, with and without a CFD solution, to illustrate the flexibility and the accuracy of the proposed automated h- and r- adaption framework designed for immersed methods.

This paper is organized as follows. First, the Immersed Method is recalled in section 2 with a description of the computation of the signed distance function. In section 3, the anisotropic mesh generation is explained in details, followed by the geometrical adaptation developed and the algorithms allowing its implementation in order to create an anisotropic fitted mesh for immersed geometries (Section 4). Section 5 describes the developed variational multiscale solver, and section 6 presents 2D test cases illustrating the procedure. Finally, section 7 is dedicated to conclusions and perspectives.

2 The Immersed Method

The Immersed Method consists of solving a multi-component system using a monolithic formulation based on a *level-set approach* without any need for a forcing model. This method relies on the use of an anisotropic mesh to adapt the interface between the two components [1, 11]. The general principle of this technique is as follows:

1. The immersed geometry is described by a signed distance function ϕ(level-set function).
2. The physical properties of each domain are unified.

2.1 Level-set Function

A level-set or signed distance function ϕ of an interface Γ is used to determine the position of the interface of the immersed body. The immersed interface is useful because of its relative simplicity and computational efficiency compared with the body-fitted. For any point x^p in the domain, the level-set function corresponds to the shortest distance to the interface Γ. This is achieved by determining the elements with the minimum surface distance, which is the shortest distance value from the surface to x^p with $|\mathbf{d}| = min_{e=1}(|\mathbf{d_e}|)$.
In order to find the minimum distance $|\mathbf{d_e}|$ in each element a bounding box check is done. It determines if their exists another candidate element closer to fx^p based on the current $|\mathbf{d}|$. If x^p is outside of the bounding box then the current minimum $|\mathbf{d}|$ remains however if it lies inside it, a projected volume is done.
The general idea of projected volume check consists of determining whether x^p lies inside or outside the projected volume of an element:

- if x^p is inside, then $|\mathbf{d_e}|$ is taken as the shortest distance between x^p and the plane defined by the element face normal $|\mathbf{n_e}|$. The condition where this is satisfied is: $(x^{1p} \cdot \mathbf{n_3} \leqslant 0) \cap (x^{2p} \cdot \mathbf{n_1} \leqslant 0) \cap (x^{3p} \cdot \mathbf{n_2} \leqslant 0)$, where $x^{ij} = x^j - x^i$ and $\mathbf{n_1}, \mathbf{n_2}$ and $\mathbf{n_3}$ are the edge normals.

- If x^p lies outside the projected volume of the element, each edge of the triangle is examined to get the shortest distance to any of the line segments $|\mathbf{d_e}|$.

Finally, the level-set function $\phi = |\mathbf{d}| \frac{x_{pc} \cdot \mathbf{n_e}}{|x^{pc} \cdot \mathbf{n_e}|}$, where $x^{pc} \cdot \mathbf{n_e}$ will be positive inside the immersed geometry Ω_{solid} and negative in the rest of domain [12]:

$$\begin{cases} \phi(x) = d(x, \Gamma) & \text{for} \quad x \in \Omega_{solid} \\ \phi(x) = -d(x, \Gamma) & \text{for} \quad x \notin \Omega_{solid} \end{cases} \tag{1}$$

The interface of the immersed object is determined by the zero-value of ϕ (Figure 1).
Further details on how to compute the signed distance function can be found in [20, 23, 36, 39].

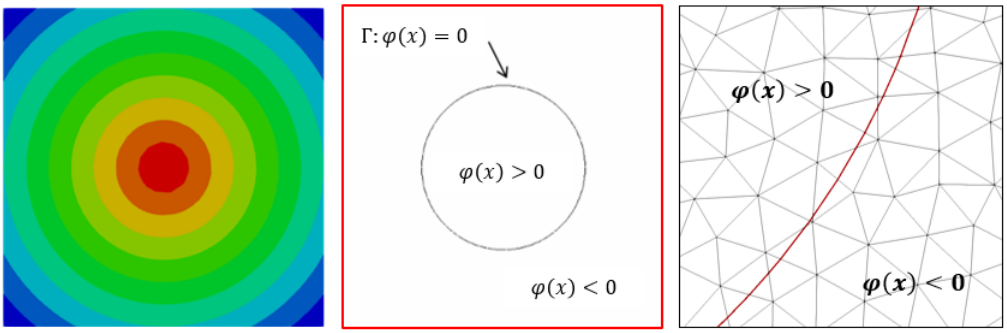


Figure 1: A level set function (*in red*) for a circle separating two domains.

In order to decrease the computational time cost needed to compute all $|\mathbf{d_e}|$, a hierarchical representation of the surface mesh is done [20, 21, 22]. A box tree is created by dividing the domain into levels of small boxes where the lowest level contains the entire mesh. Children boxes are recursively created by packing the elements of the mesh and determining the parent box that contains elements of the immersed interface or surface elements. Hence, the distance between x^p and a box $\mathcal{C}_i$ is evaluated before computing the distance between x^p and the children of $\mathcal{C}_i$ or its elements.

2.2 Unified Physical Properties

In order to account for the physical properties of the two subdomains, the Mixing law is used, along with a smoothed Heaviside function applied over a narrow band ϵ. This allows to achieve a better continuity at the

interface for the fluid-solid mixture and treat the two materials as one composite domain:

$$H(x) = \begin{cases} 1 & \text{for } \phi(x) > \epsilon \\ \frac{1}{2}(1 + \frac{\phi(x)}{\epsilon} + \frac{1}{\pi}\sin(\frac{\pi\phi(x)}{\epsilon})) & \text{for } |\phi(x)| \leq \epsilon \\ 0 & \text{for } \phi(x) < -\epsilon \end{cases} \tag{2}$$

$$\rho = \rho_{fluid} H(\phi(x)) + \rho_{solid}(1 - H(\phi(x)) \tag{3}$$

$$\mu = \mu_{fluid} H(\phi(x)) + \mu_{solid}(1 - H(\phi(x)) \tag{4}$$

where ρ_{fluid}, μ_{fluid}, ρ_{solid} and μ_{solid} are the densities and dynamic viscosity of the fluid and solid, respectively.

Through these two steps, the Immersed Method describes the immersed geometry using the level-set method and applies the physical and thermodynamic properties on either side of the immersed interface. However as seen in Figure 1, the interface of the immersed geometry intersects the elements of the mesh.

3 Anisotropic Mesh adaptation

Anisotropic mesh adaptation is a powerful method that not only increases the quality of the mesh and the accuracy of the numerical solution but also reduces significantly their CPU time. The domain discretization is accomplished following the size and directional constraints, hence, mesh elements are concentrated in regions with high gradients and large discontinuities allowing their capture with higher accuracy. The anisotropic mesh algorithm allows also to control the number of nodes and to get the smallest error possible, therefore the algorithm is designed to build the mesh, compute the numerical solution and evaluate an estimation of the interpolation error. To obtain the optimal metric, an edge based error estimator and a gradient recovery procedure are defined. The mesh is then generated according to the new metric field [33, 27].

3.1 Edge-based error estimation

Let u_h be a $\mathbb{P}_1$ finite element approximation obtained by applying the Lagrange interpolation operator $\mathbf{\Pi}$ to a function $u \in C^2(\Omega)$ and $x = \left\{x^i \in \mathbb{R}^d, i = 1, ...N\right\}$ the set of vertices in the mesh. For each node i of the mesh, let $U_i = u(x^i) = u_h(x^i)$ and $\Gamma(i)$ the set of vertices connected to x^i by a common edge x^{ij} such in Figure 2, such that $x^{ij} = x^j - x^i$ and $U^{ij} = U^j - U^i$.

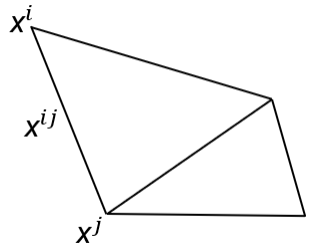


Figure 2: The set of nodes connected to x^i, and the common edge x^{ij} joining the nodes i and j.

The gradient $\nabla u^h \cdot x^{ij}$ on the edge x_{ij} is continuous, therefore we can write

$$U^j = U^i + \nabla u^h \cdot x^{ij} \tag{5}$$

which leads to

$$\nabla u_h \cdot x^{ij} = U^j - U^i \tag{6}$$

From [27], the error estimator can be defined as:

$$\| \nabla u^h \cdot x^{ij} - \nabla u(x^i) \cdot x^{ij} \| \leq \max_{y \in |x^i, x^j|} | x^{ij} \cdot H_u(y) \cdot x^{ij} | \tag{7}$$

with H_u the Hessian of u. At the node x^i, the gradient g^i of u_h can be written as:

$$\nabla g_h \cdot x^{ij} = g^j - g^i \tag{8}$$

Hence, the projection of the Hessian based on the gradient at the extremities of the edge is:

$$(\nabla g_h \cdot x^{ij}) \cdot x^{ij} = (g^j - g^i) \cdot x^{ij} \tag{9}$$

$$(H_u \cdot x^{ij}) \cdot x^{ij} = g^{ij} \cdot x^{ij} \tag{10}$$

with $g^{ij} = g^j - g^i$. From [27], it can be shown that $| g^{ij} \cdot x^{ij} |$ gives a second order accurate approximation of the second derivative of u along the edge x^{ij}. The error along the edges can then be defined as:

$$e^{ij} = | g^{ij} \cdot x^{ij} | \tag{11}$$

Equation 11 is the exact interpolation error along the edge and allows the evaluation of the global L_1 error. However, the interpolation error can only be evaluated when the gradient at the vertices is known and continuous on the nodes, thus a recovery method needs to be considered.

3.2 Gradient recovery procedure

The gradient recovery operator is defined by a local optimization problem:

$$G^i = arg \min_G \left(\sum_{j \in \Gamma(i)} | (G - \nabla u_h) \cdot x^{ij} |^2 \right) \tag{12}$$

where G^i is the recovered gradient. Let X^i be the length distribution tensor at node i defined as:

$$X^i = \frac{1}{| \Gamma(i) |} \left(\sum_{j \in \Gamma(i)} x^{ij} \otimes x^{ij} \right) \tag{13}$$

The recovered gradient G^i can then be expressed in terms of the length distribution tensor as follows:

$$G^i = (X^i)^{-1} \sum_{j \in \Gamma(i)} U^{ij} x^{ij} \tag{14}$$

and finally, the estimated error e_{ij} is written as:

$$e_{ij} = G^{ij} . x^{ij} \tag{15}$$

3.3 Metric generation

In order to link the error variation to the changes in the length of the edges, a stretching factor s^{ij} is introduced. However, it's insufficient to only enlarge the edge if the error is too small or to reduce it or break it if it is too large, the neighborhood of the node must be taken into account: a metric is thus the best averaging representation. It is defined as follows:

$$\widetilde{M^i} = (\widetilde{X}^i)^{-1} \tag{16}$$

where

$$\widetilde{X}^i = \frac{1}{\mid \Gamma(i) \mid} \left(\sum_{j \in \Gamma(i)} s^{ij} \otimes s^{ij} \right) \tag{17}$$

The stretching factor s^{ij} of the edge ij is chosen so that the total number of nodes in the mesh is kept fixed and is defined as

$$s^{ij} = \left(\frac{e_{ij}}{e(N)} \right) \tag{18}$$

where $e(N)$ the total error.

3.4 Mesh adaptation

3.4.1 Criteria

In a multi-component application, the interface between the solid and the liquid need to be modeled accurately. Hence, the mesh should be adapted to several variables, like the velocity and the level-set function. The most common way is to define a metric for each variable and combine them using metric intersection operation. However, in this work, this operation is simplified and one metric is used to account for the changes in all variables. Equation 11 is extended to consider all sources of error defined in vector $\mathbf{v}(x^i)$:

$$\mathbf{v}(x^i) = \left\{ \frac{V^i}{\mid V^i \mid}, \frac{\mid V^i \mid}{max_j \mid V^j \mid}, \frac{\phi}{max(\phi)} \right\} \tag{19}$$

Note that since all fields are normalized, the variations of all the variables are taken into account.

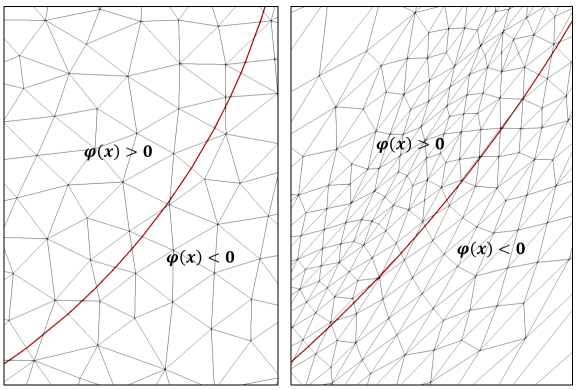


Figure 3: A Level-set function computed on an isotropic *(left)* and anisotropic *(right)* meshes.

Figure 3 compares the interface captured by an isotropic and anisotropic mesh for the same immersed geometry. Applying the anisotropic mesh adaptation on the level-set function, the elements surrounding the interface are stretched along its direction and hence define it more precisely. However, the interface remains intersect with the elements of the mesh. To capture accurately capture the interface and create an anisotropic body-fitted mesh, a new geometrical adaptation method is proposed.

3.4.2 Advantages

Since CFD and fluid flow problems use complex geometries, high resolution CAD models (or STL) require to be treated. The use of immersed geometries and how they are expressed on the interface depends on the end use and the level of geometric information needed at the interface. Many accurate interface description using

high-order polynomial representations [43] or level set functions [44, 45] have been used. The mathematical structure of the governing equations being solved frequently influences the augmentation of discrete operators employing immersed interfaces, which are divided into incompressible [46] or compressible formulations [47, 48]. Some methods, like the ghost fluid method, often requires one or more of the following steps:

1. inserting local nodes at the interface
2. increasing the size of the discrete stencils near the interface
3. adding nodes in the solid region in the vicinity of the interface to enforce local boundary conditions

Other methods rely on transforming the background mesh to conform to the boundary by using a closest point projection to parameterize the immersed boundary over a collection of nearby edges [42], or on augmenting the level-set function via an iterative procedure to then reconstruct the cut cell elements using Lagrangian polynomials [23].

Since the first step of the proposed method is to apply an anisotropic mesh adaptation, the mesh is locally refined according to the level-set function that describes the geometry without resorting to the reconstruction or correction of the interface from a CAD model. The anisotropic mesh adaptation gives the flexibility to adapt to the immersed geometry and to construct the stretched elements in the direction of the interface allowing to smoothly track the curve of the geometry. Unlike isotropic elements, the smoothness of the immersed interface and it's detection can be done with a relatively low number of elements as can be seen in Table 1.

The metric map allows an adaptation on the level-set function as well as the solution of the problem as mentioned in section 3.4.1, the boundary layer is then well constructed taking onto account the flow physics. The Dirichlet boundary conditions can then be strongly imposed on the interface just like classical body-fitted meshes.

4 Anisotropic Adaptive Fitted Mesh

Creating a body-fitted mesh for an immersed geometry is a challenge. In this section, the general principle of the new adaptation technique, allowing the capture of the boundary between the two components, is described. It's based on capturing the interface following a **geometrical adaptation**. Geometry based adaptation consists of modifying the elements of the mesh:

- by relocating mesh nodes to some regions of interest to better capture the physical and mechanical properties desired - this is known as **R-adaptation** (Section 4.1),
- by introducing new nodes in the region of interest hence creating new elements which requires remeshing. The introduced nodes are the intersection points between two nodes i and j, and are computed by linear interpolation using the following equation:

$$\mathcal{X}^{ij} = \mathcal{X}^i - (\mathcal{X}^i - \mathcal{X}^j)\frac{\phi(\mathcal{X}^i)}{\phi(\mathcal{X}^i) - \phi(\mathcal{X}^j)} \tag{20}$$

- by conserving the number of elements but inducing a change in the overall topology of the mesh like in **edge-swapping** or edge-flipping [15] (Section 4.1).

4.1 Geometric Adaptation for a fitted mesh

The steps of the proposed technique are illustrated in the diagram of figure 4. Algorithm 1 summarizes the overall technique and the details are explained below (Algorithms 2 to 4).

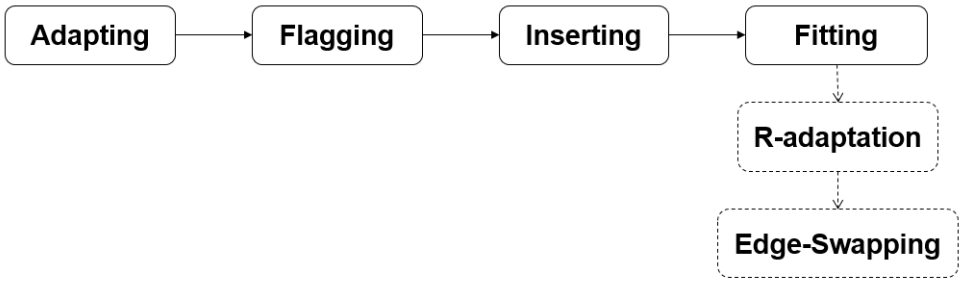


Figure 4: Scheme for applying the Geometric Adaptation.

Algorithm 1 Geometric Adaptation for a fitted mesh

```
Apply anisotropic adaptation on the level-set.
Flag the cut elements where φ_i φ_j < 0 with i and j nodes of an edge
for each flagged element do
    Move the node with the minimum distance
Compute couples with one fitted node
for each couple do
    Apply Edge Swapping keeping in mind the orientation of the elements
```

- **Adapting**

After generating an isotropic mesh for the immersed geometry, an anisotropic mesh adaptation is applied to the level-set using the metric map described in equation (16). On a narrow band region in the vicinity of the interface, extremely anisotropic elements stretched along the boundary are formed. The anisotropic refinement allows us to follow the geometric shape of the immersed geometry hence describing its curvatures, angles, direction, etc. (Figures 5).

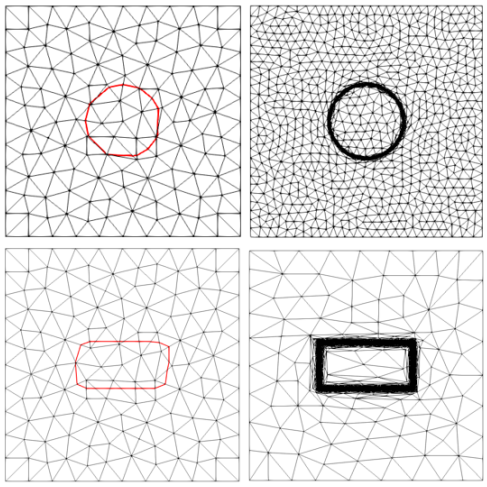
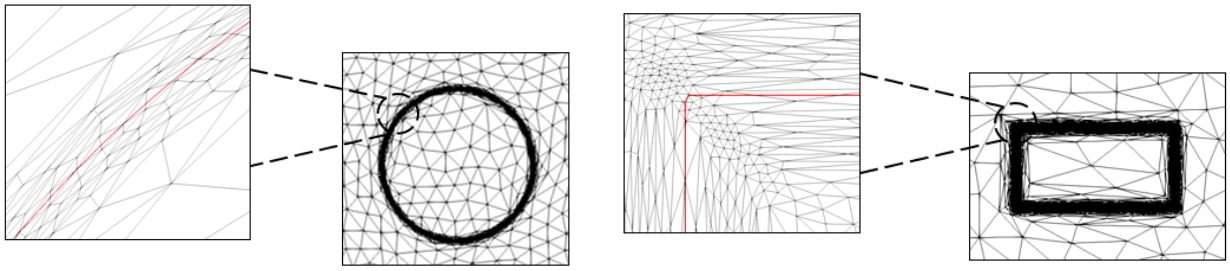


Figure 5: Anisotropic adaptation for 2D geometries.

- **Flagging**

Since the immersed geometry is described using the level-set method, the interface Γ separates the domain



(a) (b)

Figure 6: Zoom on the interfaces of a cylinder and a rectangle highlighting the cut elements.

Ω into two subdomains: a positive one and a negative one [38]. Therefore, the interface elements or the *cut elements* are detected by looping over their edges. The product of the signed distance value at each node (i, j) of a cut edge results in a negative value allowing the detection of the cut elements (Algorithm 2 and figure 7).

Algorithm 2 Flagging

```
for each edge (ij) do
    Compute φ_i and φ_j
    if φ_i φ_j < 0 then
        flag = 1                          ▷ Highlighting the cut element
    else
        flag = 0
flag(i), i ∈ {1, ..., N}      N: # of nodes
```

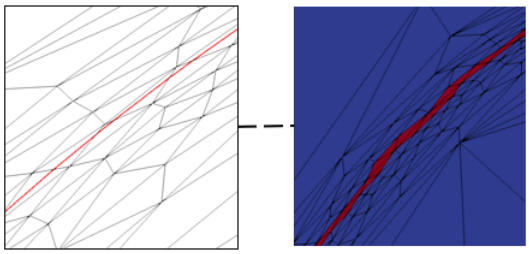


Figure 7: The flagged elements cut by the interface.

• Inserting & Fitting

▷ R-adaptation

R-adaption is based on relocating the nodes by maintaining the same number of degrees of freedom and element connectivity. It's applied here only on the cut elements, as a consequence, the computational cost remains low.

The general strategy consists of moving specific nodes of the mesh. For each cut element, the distance from each node to the interface is analyzed via the level-set function: the node having the minimum distance v_{m_i} is then marked. Since the interface cuts the facets of the mesh: the intersection points are then defined as a virtual nodes v_f such as $f = 0, 1, ...n_f$ with n_f the number of cut facets.

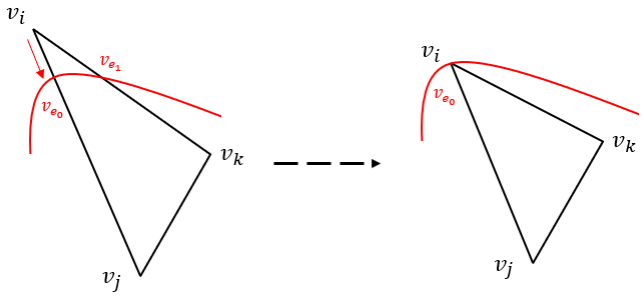


Figure 8: R-adaptation operation for two 2D cut elements applied on node v_i.

The objective is to move the marked node v_{m_i} along its associated facet f with minimum effort. Since v_{m_i} can be associated to multiple virtual nodes, a set of associated virtual nodes is created and analyzed for each v_{m_i}. From this set, v_f is chosen as the virtual node having the smallest distance $\mathbf{d_{v_{m_i} v_f}}$, then the coordinates $\mathcal{X}_{m_i}$ of v_{m_i} are updated as follows:

$$\mathcal{X}_{m_i} = \mathcal{X}_{v_f} \tag{21}$$

with $\mathcal{X}_{v_f}$ the coordinates of the chosen virtual node associated to v_{m_f}.

Algorithm 3 R-adaptation

1: **for** the nodes of a cut element **do**
2: Mark the node having minimum distance
3: **for** each marked node **do**
4: Determine the needed virtual node v_f of the associated facet f and the interface
5: $\mathcal{X}_{inew} = \mathcal{X}_{v_f}$
6: $\mathcal{X}(i), i \in \{1, \ldots, n\}$ n: # of marked nodes in cut elements

To prevent mesh degeneration or inverted elements, care must be taken. Hence, constraints on the number of nodes marked are applied: each cut element should have at least one node marked and no element has all of its nodes marked. And each v_f can be associated to only one real marked node v_{m_0}.

▷ **Edge Swapping**

The R-adaptation algorithm allows the movement of the nodes' cut elements to capture the interface. To ensure a fitted mesh, algorithm 3 is coupled with a local mesh optimization algorithm. This algorithm is based on a simple topology change method: Edge swapping. Edge swapping not only avoids expensive remeshing but preserves the quality and integrity of the mesh. This operation conserves the number of elements and the number of edges. In 2D, this local optimization is a simple topological operation consisting of swapping or flipping a common edge shared by two elements (forming a pair or a couple) (Figure 9).

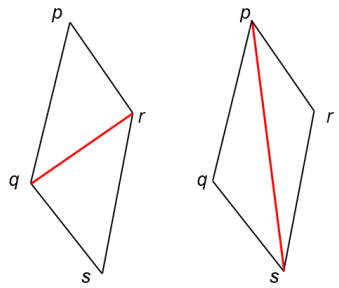


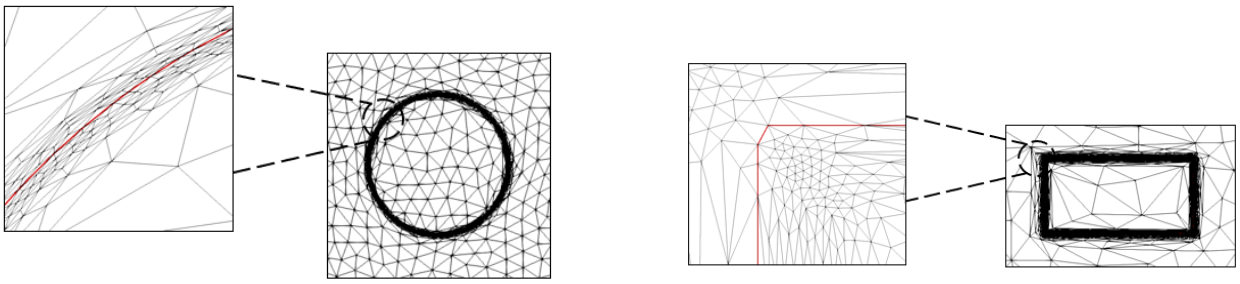
Figure 9: Edge swapping operation in 2D.

Since edge-swapping alters the topology of the mesh, a careful evaluation of the global connectivity of the mesh needs to be done to ensure that the orientation of the elements is preserved [13]. Therefore, the edge swapping algorithm can be decomposed into three main parts:

1. Pairing the elements
2. Swapping
3. Fixing the orientation

Algorithm 4 Edge swapping

```
Loop over the cut elements
Flag the elements with only one node on the interface
Loop over the newly flagged elements
if two elements have a common edge then
    Form a couple                                   ▷ Pairing the elements
for each couple do
    Determine the common edge
    Swap the edge                                   ▷ Edge-Swapping
    for each element do
        Calculate the signed volume 𝒱
        while 𝒱 < 0 do
            Reorder the nodes indices               ▷ Fixing the orientation
```



(a) (b)

Figure 10: Zoom on the fitted interface for a cylinder and a rectangle.

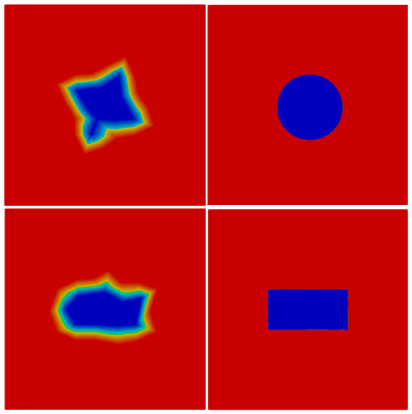


Figure 11: Comparison between the initial and final fitted geometry.

The geometric adaptation can be used to solve multi-component systems with complex geometries, CFD problems and Fluid-Solid Interaction applications. Before presenting some examples and numerical applications in section 6, we describe in section 5 the numerical resolution of the Navier-Stokes equations.

5 Numerical resolution of the Navier-Stokes equations

Let $\Omega \subset \mathbb{R}^d$, with d the space dimension, and $d\Omega$ its boundary. We consider the following velocity $\mathbf{u}$ - pressure p formulation of the Navier-Stokes equations for unsteady incompressible flows:

$$\begin{cases} \rho(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u}) - \nabla \cdot \sigma = f \\ \nabla \cdot \mathbf{u} = 0 \end{cases} \tag{22}$$

where ρ is the density, f the body force vector per unity density and σ the stress tensor such as:

$$\sigma = 2\mu\epsilon(\mathbf{u}) - p\mathbf{I_d} \tag{23}$$

with μ the dynamic viscosity, $\mathbf{I_d}$ the identity tensor, and ϵ the strain-rate tensor defined as:

$$\epsilon(\mathbf{u}) = \frac{1}{2}(\nabla \mathbf{u} + \nabla \mathbf{u}^T) \tag{24}$$

To prevent spurious oscillations resulting from the convection-dominated regimes and solve the pressure instability problem, a variational multiscale method for the Navier– Stokes equations is used [25, 26, 18]. The weak formulation of (22)-(23) with velocity space $V \subset [H^1(\Omega)]^d$ and pressure space $Q = \{q \in L^2(\Omega) : \int_\Omega q = 0\}$ consists in finding $(u, p) \in V \times Q$ such that:

$$\begin{cases} ((\rho(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u})), \mathbf{w}) + (2\mu\epsilon(\mathbf{u}) : \epsilon(\mathbf{w})) - (p, \nabla \cdot \mathbf{w}) = (\mathbf{f}, \mathbf{w}), & \forall \mathbf{w} \in V \\ (\nabla \cdot \mathbf{u}, q) = 0, & \forall q \in \Omega \end{cases} \tag{25}$$

with $(.,.)$ the L^2 inner product over Ω.

Let $\tau_\mathbf{h}$ be an admissible mesh constructed as a triangulation of Ω, and V_h and Q_h the finite dimensional spaces approximations of the function spaces V and Q, respectively. To ensure the stability of (25), the choice of V_h and Q_h must fulfill a compatibility condition [16].

In this work, $\mathbb{P}_1/\mathbb{P}_1$ element with a Variational Multiscale method (VMS) [17] is used to ensure the stablization. All unknowns are divided into a coarse and a fine component. The fine scales are solved in an approximate manner and modeled in function of the residual based terms. Their effect is then transferred into the large scale equations. The coarse and fine components of the velocity and pressure fields are:

$$\mathbf{u} = \mathbf{u}_h + \mathbf{u}' \tag{26}$$

$$p = p_h + p' \tag{27}$$

as well as their respective weight functions:

$$\mathbf{w} = \mathbf{w}_h + \mathbf{w}' \tag{28}$$

$$q = q_h + q' \tag{29}$$

The enriched function spaces are defined as $V = V_h \oplus V', V_0 = V_{h,0} \oplus V'$and $Q = Q_h \oplus Q'$. Therefore, the resulting finite element approximation of the time-dependent Navier–Stokes problem consists in finding (u, p) $inV \times Q$ such that:

$$\begin{cases} (\rho(\partial_t(\mathbf{u}_h + \mathbf{u}') + (\mathbf{u}_h + \mathbf{u}') \cdot \nabla(\mathbf{u}_h + \mathbf{u}')), (\mathbf{w}_h + \mathbf{w}')) \\ +(2\mu\epsilon(\mathbf{u}_h + \mathbf{u}') : \epsilon((\mathbf{w}_h + \mathbf{w}'))) - ((p_h + p'), \nabla \cdot (\mathbf{w}_h + \mathbf{w}')) \\ = (\mathbf{f}, (\mathbf{w}_h + \mathbf{w}')), \quad \forall \mathbf{w} \in V_0 \\ (\nabla \cdot (\mathbf{u}_h + \mathbf{u}'), (q_h + q')) = 0, \qquad \forall q \in Q \end{cases} \tag{30}$$

The stabilized formulation is then derived from equation (30) by forming fine and large scale problems. The fine-scale problem is defined on element interiors, and $\mathbf{u}'$ and p' are written in terms of the time-dependent large-scale variables using consistently derived residual-based terms.Then, $\mathbf{u}'$ and p' are directly replaced into the large-scale problem, which gives rise to additional terms in the Finite Element formulation, tuned by a local stabilizing parameter. These terms are responsible for the enhanced stability compared to the standard Galerkin formulation. Finally, the coarse-scale equations can be computed:

$$\begin{cases} (\rho(\partial_t(\mathbf{u}_h + \mathbf{u}_h \cdot \nabla \mathbf{u}_h), \mathbf{w}_h) - (\tau_1 \mathcal{R}_m, \rho \mathbf{u}_h \cdot \nabla \mathbf{w}_h) + (2\mu\epsilon(\mathbf{u}_h) : \epsilon(\mathbf{w}_h)) \\ -(p_h, \nabla \cdot \mathbf{w}_h) - (\tau_2 \mathcal{R}_c, \nabla \cdot \mathbf{w}_h) = (\mathbf{f}, \mathbf{w}_h), \qquad \forall \mathbf{w_h} \in V_{h0} \\ (\nabla \cdot \mathbf{u}_h, q_h) - (\tau_1 \mathcal{R}_m, \nabla q_h) = 0, \qquad \forall q_h \in Q_h \end{cases} \tag{31}$$

with $\mathcal{R}_m$ and $\mathcal{R}_c$ are pieciewise constant momentum and continuity residuals:

$$\mathcal{R}_m = \rho(\partial_t \mathbf{u}_h + \mathbf{u}_h \cdot \nabla \mathbf{u}_h) - \nabla p_h \tag{32}$$

$$\mathcal{R}_c = -\nabla \mathbf{u}_h \tag{33}$$

and τ_1 and τ_2 are piecewise defined stabilization parameters adopted from [24]. More details about the formulation of VMS can be found in [18].

6 Numerical tests

The geometric adaptation algorithm is applied first on a set of geometries to show the accurate and precise capture of the interface and then benchmark problems are explored. The objective of exploring these cases is to understand the impact and importance of an immersed fitted interface.

6.1 Anisotropic Fitted mesh for immersed 2D objects

In order to confirm the results, the global adaptive framework was implemented successfully on simple geometries (Figures 5 - 11) as well as the geometry of a rabbit (Figures 12-14) with different curvatures. First, an anisotropic adaptation, implemented on the level-set function of the immersed geometry, is applied. The use of anisotropic elements is an essential component to accurately trace the different changes in the geometry due to the flexibility of such elements (Figure 12).

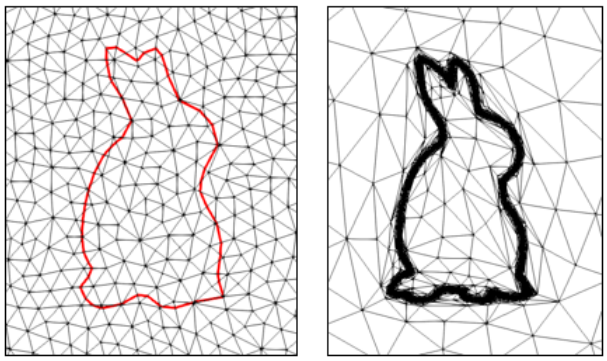


Figure 12: Anisotropic adaptation on the geometry of a 2D rabbit.

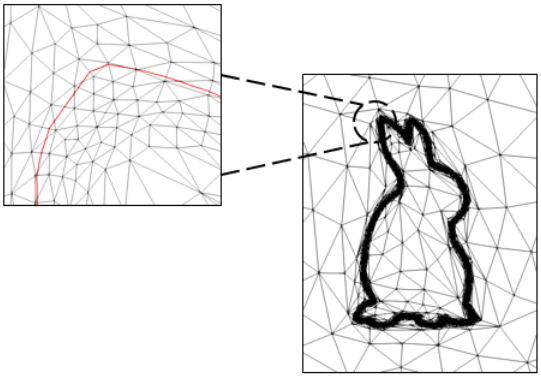


Figure 13: Zoom on the interface highlighting the cut elements

Then the immersed-fitting adaptation was applied to the elements in the vicinity of the level-set, represented in red in figures 13 and 14. The R-adaptation relocates the marked nodes to get them closer to the interface, and then edge-swapping is applied to ensure a body-fitted mesh.

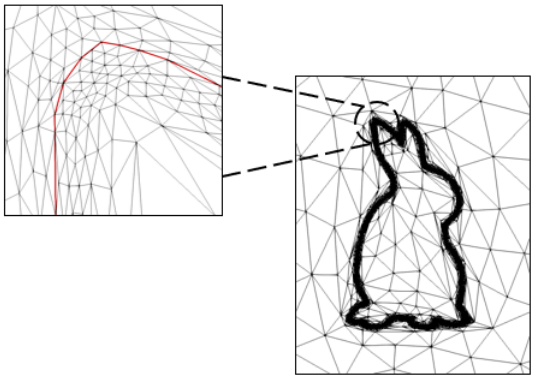


Figure 14: Zoom on the fitted interface.

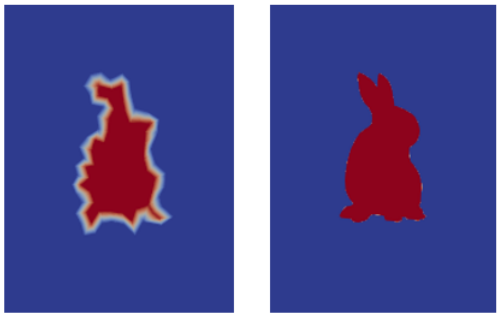


Figure 15: Comparison between the initial and the final fitted mesh of a 2D rabbit.

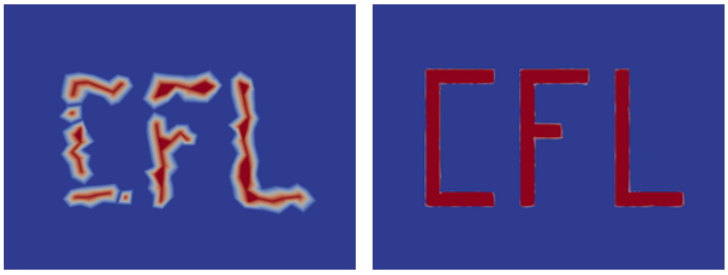


Figure 16: Comparison between Isotropic *(left)* and Immersed-Fitted mesh *(right)*.

Figures 15 and 16 show the flexibility and versatility the fitting algorithm. Moreover, the flow over four circular cylinders at various Reynolds numbers (Figure 17) was studied. Using the method presented, an anisotropic fitted mesh was successfully created on all four cylinders independent on the flow's turbulence.

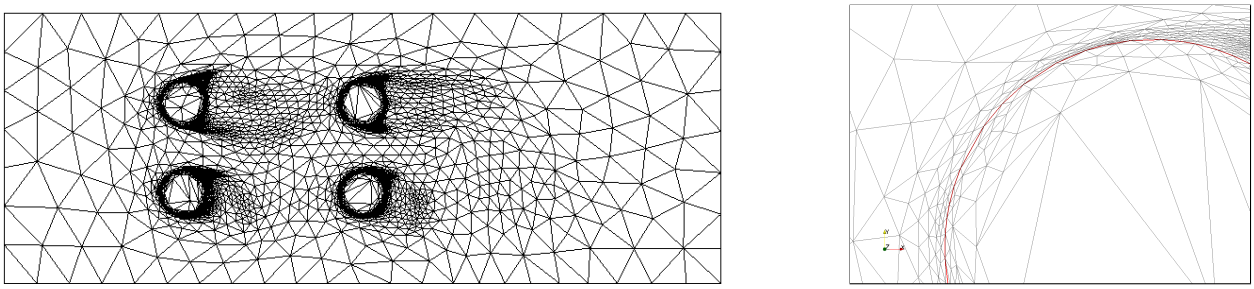


Figure 17: Immersed Fitted mesh for 4 circular cylinders and zoom on the interface.

6.2 Flow over a Circular Cylinder

We consider the flow over a circular cylinder, also studied by Main et al. in [6], at Reynolds number $Re = 100$ over which the Navier-Stokes equations for incompressible flow are solved using the Variational Multiscale Method.

The velocity inlet of the flow is set to $U = 1$ and the top and bottom of the domain are set to be traction free. On the outlet, a free Newman boundary condition is imposed. The Drag coefficient is calculated using the following equation:

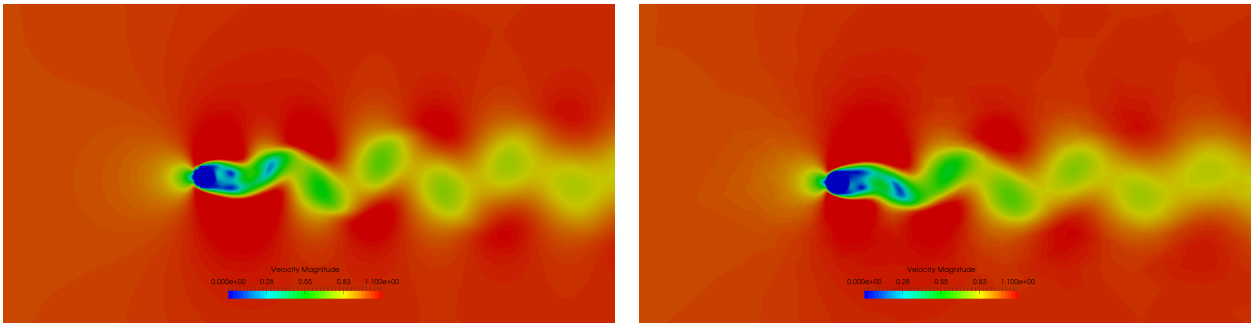
$$C_D = \frac{2F_d}{D\rho U^2} \tag{34}$$

where F_d is the drag force.

Table 1 compares the drag coefficient obtained for three types of meshes:

1. Isotropic initial mesh
2. Anisotropic mesh
3. Immersed-fitted mesh

The immersed-fitted mesh was obtained by implementing the algorithm defined in Section 4 by applying an anisotropic adaptation near the interface and using the Immersed Method described in section 2.



(a) (b)

Figure 18: Solution for flow past a cylinder for $Re = 100$ using an isotropic mesh $(left)$ and an immersed-fitted anisotropic mesh $(right)$ at $t = 100s$.

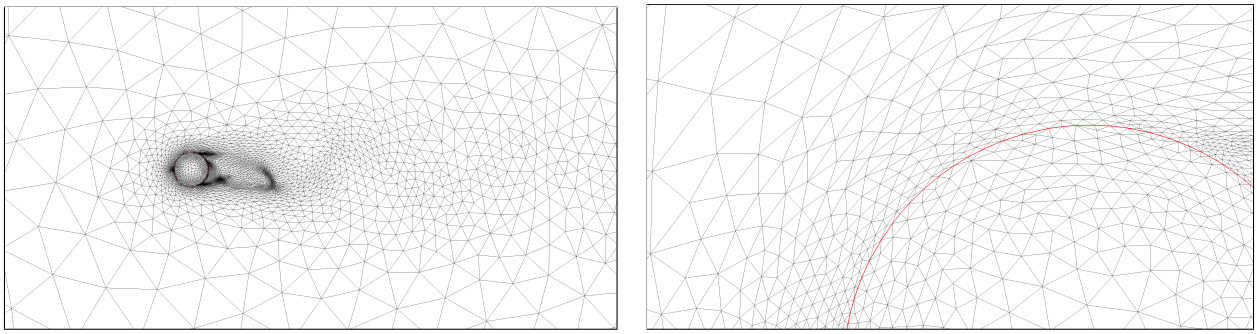


Figure 19: Anisotropic mesh adaptation for both the velocity and the level-set fields *(left)*, zoom on the sharp immersed-fitted interface *(right)*.

Table 1: Drag Coefficient C_D computed for the 3 cases for Re=100 with N being the number of elements used.

	N	C_D	% Error
Reference	352 590	1.34	-
Isotropic mesh	156 163	1.413	5.45%
Anisotropic mesh	10 000	1.316	1.77%
Immersed-Fitted mesh	10 000	1.334	0.44%

The results obtained (table 1 and graph 20) confirm that using the Immersed Fitted mesh adaptation yields to better results with a minimum error in C_D equal to 0.44%. Figure 20 shows a time history of the drag coefficient for $Re = 100$, for the three cases studied.

It is important to note from table 1 that the anisotropic adaptation allows to significantly reduce the number of elements used in order to solve the Navier-Stokes equations, compared to the use of isotropic mesh: only 10 000 elements were needed compared to 156 163 elements in the case of isotropic refinement and more than 350 000 elements in [6]. Also, the computational time to calculate the results using the anisotropic or the immersed-fitted mesh is significantly lower than the time needed to solve the problem using an isotropic mesh.

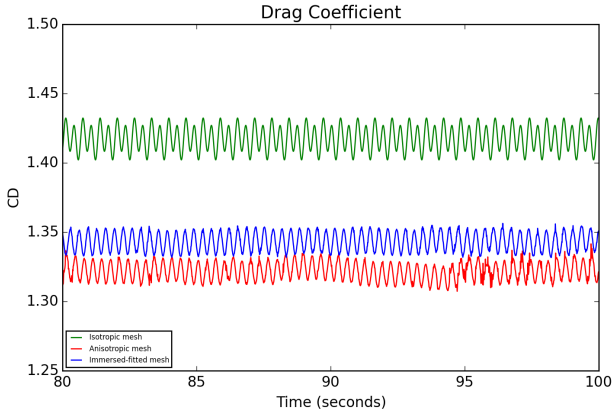


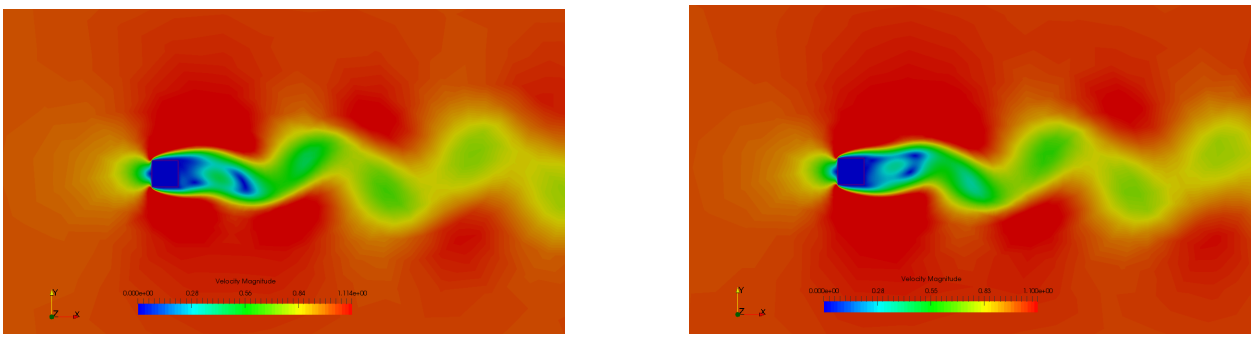
Figure 20: Evolution in time for the drag coefficient.

6.3 Flow over a Square Cylinder

We consider next a flow over a square cylinder for Re=100 presented in [19]. The computational domain consists of a square cylinder placed normal to a free stream in an infinite domain. A uniform velocity is set on the inlet $U = 1$ and no slip conditions are applied to the boundary of the immersed square.
Such a geometry was tested because its corners and sharp angles can be a challenge. The algorithm of section 4 was successfully applied to the geometry without any modifications and was able to capture the sharp edges. Table 2 compares the drag coefficient obtained for the three types of meshes: the Immersed fitted mesh shows the best results.

Table 2: Drag Coefficient C_D computed for the 3 cases for a square cylinder at Re=100.

	C_D	% Error
Reference [19]	1.461	-
Isotropic mesh	1.524	4.312%
Anisotropic mesh	1.429	2.173%
Immersed-Fitted mesh	1.433	1.926%



(a) $t = 25s$ (b) $t = 100s$

Figure 21: Solution for flow past a square cylinder for $Re = 100$ for an immersed-fitted anisotropic mesh (*bottom*) at $t = 25s$ an $t = 100s$.

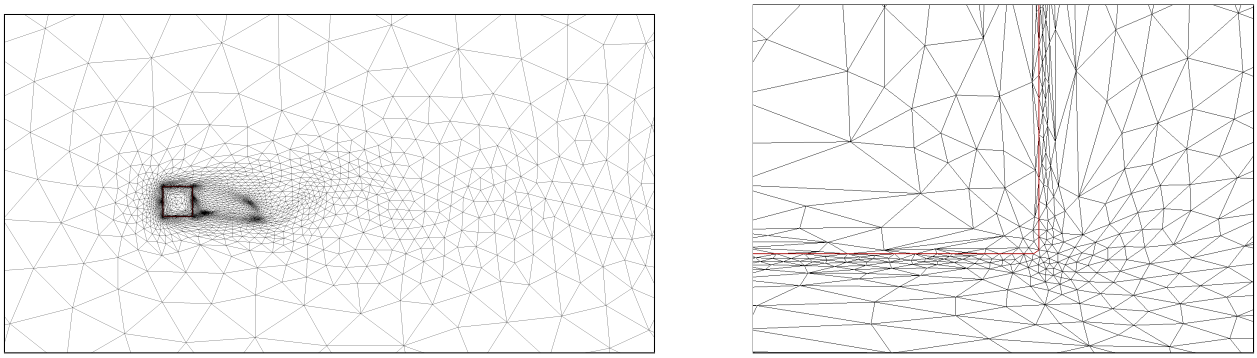
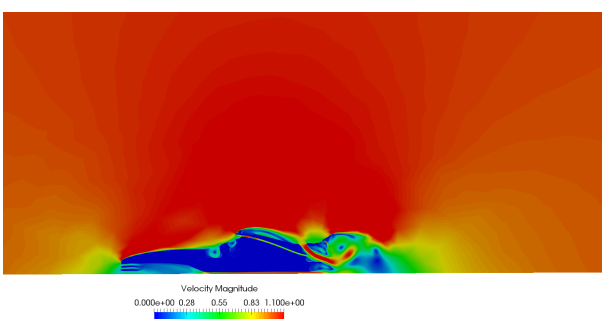


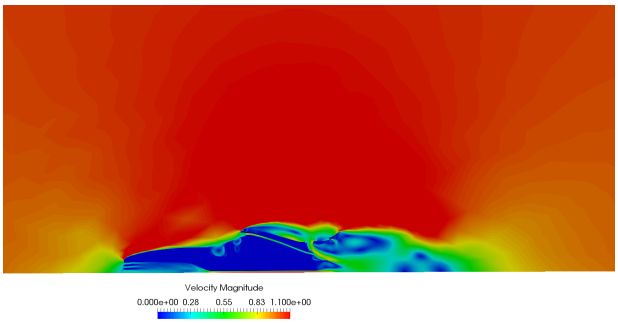
Figure 22: Anisotropic mesh adaptation for both the velocity and the level-set fields *(left)*, zoom on the sharp immersed-fitted interface *(right)*.

6.4 Complex geometry with high Reynolds number flow

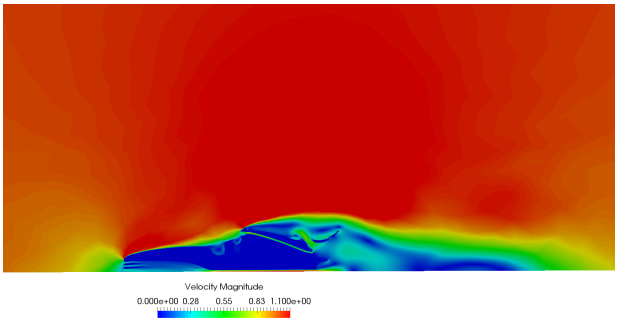
To go further, we consider the flow over a complex geometry, a 2D representation of a F1 car. The objective of this example is to show the performance of immersed-fitted meshing. Indeed, combined with flow solvers it allows to easily and accurately deal with complex fluid-structure interaction problems. Taking a closer look at the mesh near the interfaces, we can detect the good orientation of the anisotropic elements as well as the accurate precision with the conform interface. The velocity field and the mesh obtained are shown in Figures 23 and 24. Taking a closer look at the mesh in Figure 25, we can see how the interface not only follows the curvatures of the immersed geometry but also captures accurately its interface.



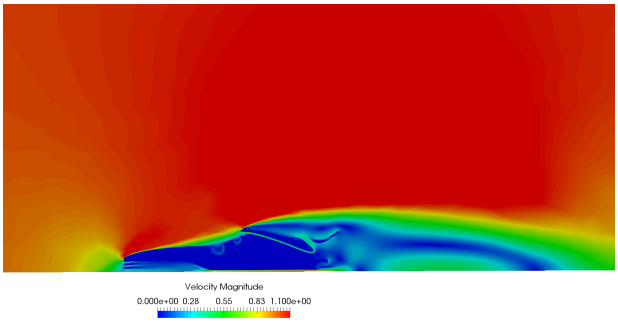
(a) $t = 2.5s$



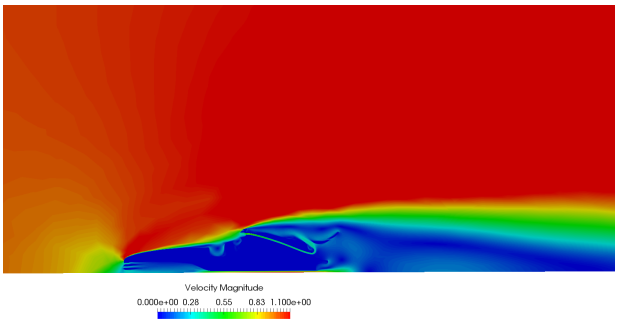
(b) $t = 5s$



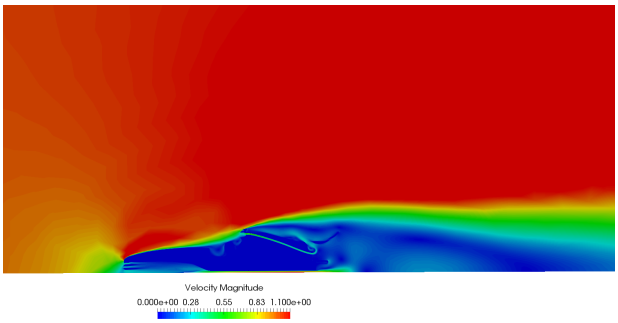
(c) $t = 10s$



(d) $t = 20s$

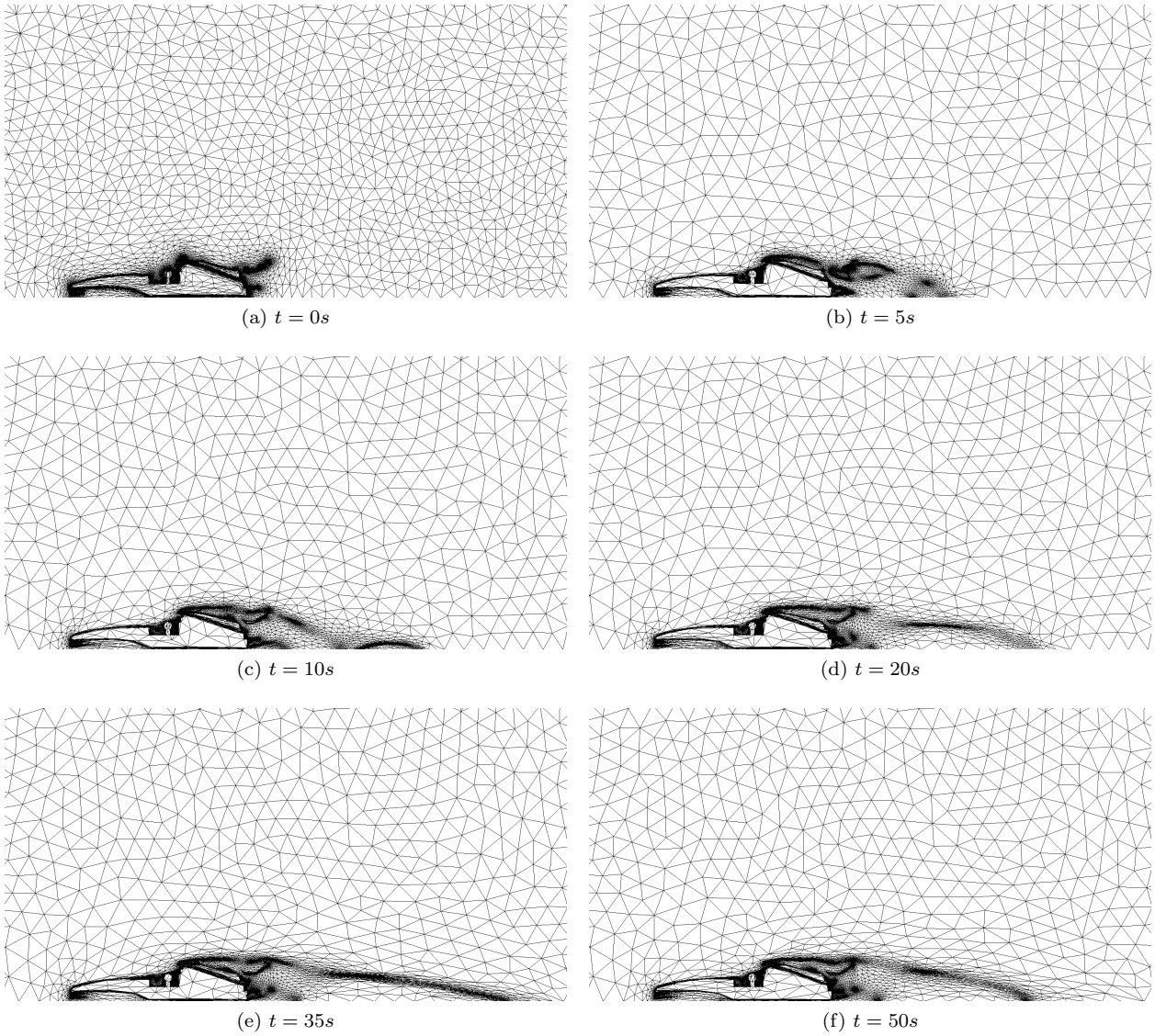


(e) $t = 35s$



(f) $t = 50s$

Figure 23: Velocity profile past an immersed 2D car.



(a) $t = 0s$ (b) $t = 5s$

(c) $t = 10s$ (d) $t = 20s$

(e) $t = 35s$ (f) $t = 50s$

Figure 24: Anisotropic mesh adaptation for both the velocity and the level-set of the immersed 2D car fields.

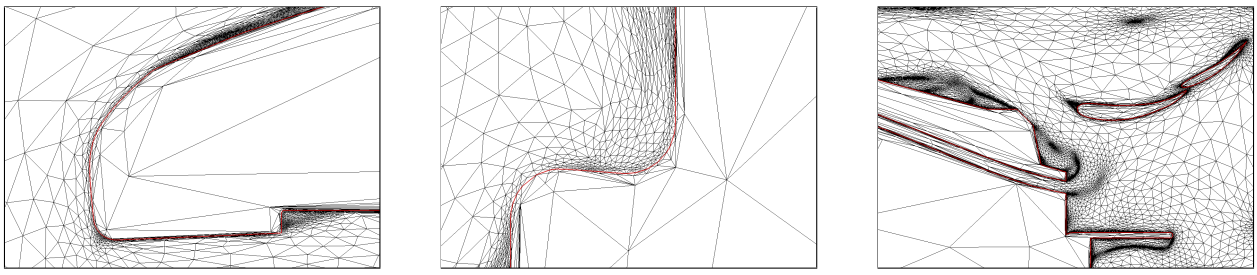


Figure 25: Zoom on the sharp immersed-fitted interface of the 2D car.

6.5 Conjugate Heat transfer application

We now consider a heat transfer application where a heated solid is immersed in a cooling cavity (Figure 26. A Cartesian coordinate system is used with origin at the center of mass of the solid that has a rectangular shape of height h and an aspect ratio of $2:1$, initially at temperature T_s. The solid is fixed at the center of a cavity of height H and aspect ratio 4:1, whose wall are isothermal and a fixed temperature T_w. Cooled air at T_c is pumped into the enclosure from two intlets located at the top of the cavity, at a velocity V_i. Two outlets for the hot air are located at the sidewalls of the cavity. Table 3 gives the numerical parameters used all expressed in SI units, and the temperature in Celsius. Using the immersed-fitted algorithm coupled with flow and heat transfer solvers, the level-set function, the velocity and the temperature are used as multi-components criteria in order to adapt the mesh on the interface of the immersed geometry as well as the solution of the equations solved. This can be illustrated in Figure 30a showing how the mesh is conform to the interface. The corresponding adapted mesh colored by the temperature distribution in Figure 30b shows how the numerical framework can capture accurately all boundary layers and shear regions via stretched elements. Figures 27 to 30 are a clear example on how the immersed-fitted algorithm evolves dynamically with the problem at hand: starting with a coarse mesh, the mesh progresses towards an anisotropic refined one without any intervention from the user. The problem is then solved simultaneously in the domain capturing the vortices formed and in the solid tracking its cooling evolution.

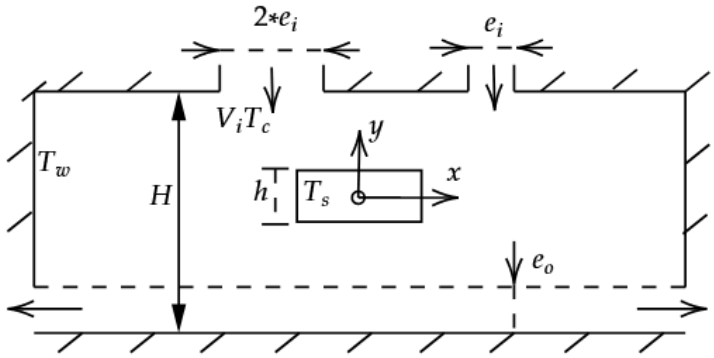
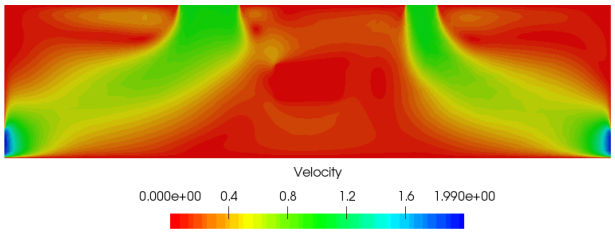


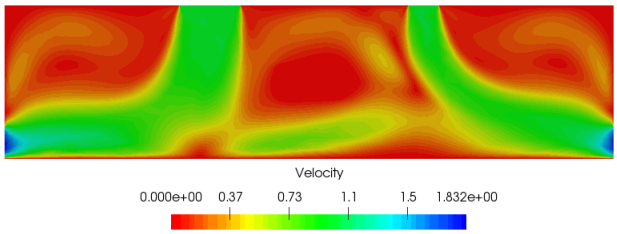
Figure 26: Schematic of the 2D forced convection set-up.

Table 3: Numerical parameters used in the 2D forced convection problem, with ρ the fluid density, μ the dynamic viscosity, λ the thermal conductivity, and c_p specific heat.

H	h	e_i	e_o	V_i	T_w	T_c	T_s	μ	ρ	λ	c_p	
1	0.2	0.2	0.2	1	10	10	150	0.001	1	0.5	1000	Fluid
								1000	100	15	300	Solid

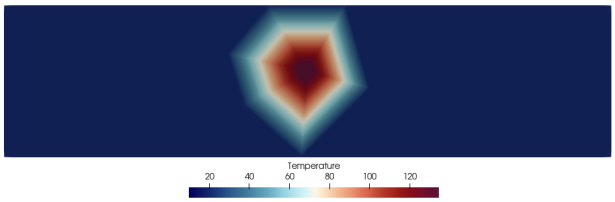


(a) $t = 1s$

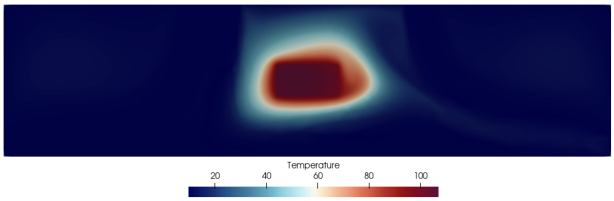


(b) $t = 150s$

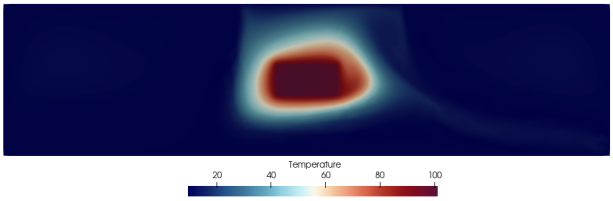
Figure 27: Velocity profiles inside the cavity.



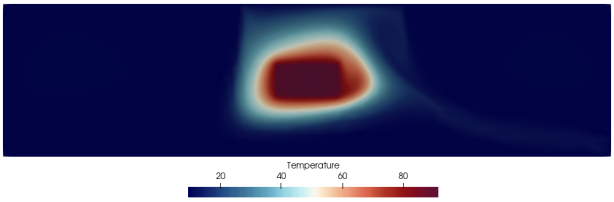
(a) $t = 0s$



(b) $t = 35s$

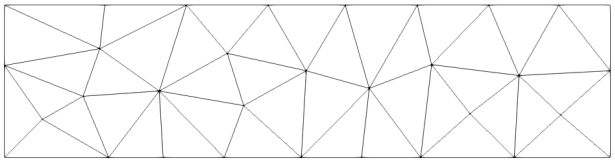


(c) $t = 70s$

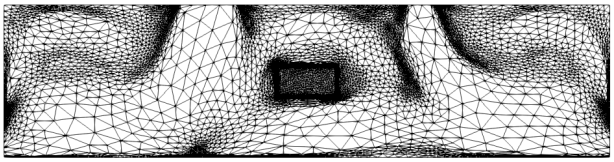


(d) $t = 150s$

Figure 28: Temperature distribution inside the cavity.

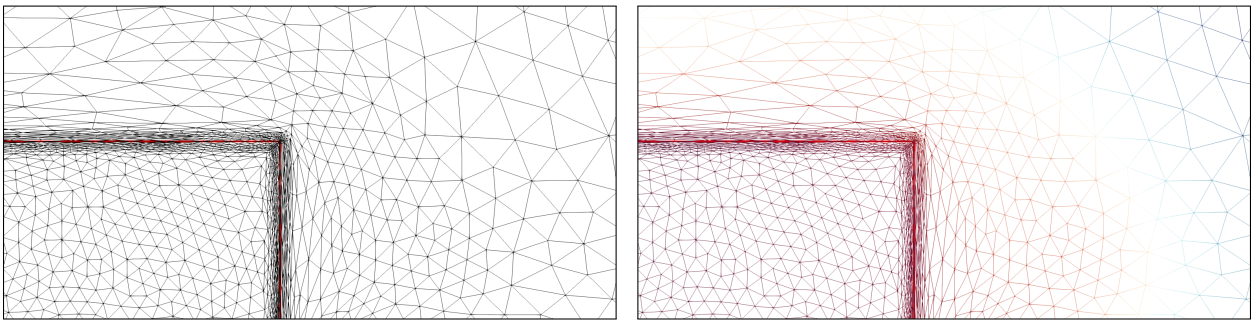


(a) $t = 0s$



(b) $t = 150s$

Figure 29: Coarse initial mesh *(left)* and the obtained anisotropic adapted one on both the velocity, temperature and the solid level-set *(right)* at $t = 150s$.



(a) (b)

Figure 30: Zoom on the sharp immersed-fitted interface of the 2D solid.

7 Conclusion

A new method is presented to create a fitted anisotropic mesh for immersed geometries. The algorithm consists on applying two main steps after isolating the cut elements: first an R-adaptation is applied to the nodes closest to the interface and then edge-swapping is applied to the remaining cut elements. This is a purely geometrical adaptation that creates a body-fitted mesh without altering the number of vertices nor elements. Combined with the anisotropic properties, not only any complex geometry can be captured but also the large gradients on the interface for any flow problem can be handled. Several 2D test cases with and without flow problems are presented and show the versatility of the method dealing with different geometries as well as its efficiency when computing the flow properties with minimum error. The algorithm presented in this work can be applied in parallel and extended to 3D applications. The extension to 3D applications present three main challenges: building a dynamic data structure to parallelize the implementation of the algorithm, updating the sparsity pattern specially during swapping since we're now dealing with faces rather than edges, and finally the main challenge is the 3D swapping. The R-adapation algorithm is followed by the swapping adaptation, here, we talk about 3D splitting rather than edge permutation: three cases of cut cells are then presented and have to be dealt with:one, two or three intersection points between the immersed surface and the elements of the mesh. This is an ongoing work that will allow to study more precisely 3D multi-component systems and their interactions.

References

[1] Elie Hachem, Hugues Digonnet, Elisabeth Massoni, and Thierry Coupez. Immersed volume method for solving natural convection, conduction and radiation of a hat-shaped disk inside a 3D enclosure, *International Journal of Numerical Methods for Heat & Fluid Flow*, 2012.

[2] Nicolas Moës, and John Dolbow, and Ted Belytschko. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering*, 46(1):131–150, 1999.

[3] Erik Burman. Ghost penalty, *Comptes Rendus Mathematique*, 348(21-22):1217–1220, 2010.

[4] Yu-Heng Tseng and Joel H Ferziger. A ghost-cell immersed boundary method for flow in complex *Journal of Computational Physics*,192(2):593-623,2003.

[5] Alex Main and Guglielmo Scovazzi. The shifted boundary method for embedded domain computations. Part I: Poisson and Stokes problems. *Journal of Computational Physics*, 372:972–995, 2018.

[6] Alex Main and Guglielmo Scovazzi. The shifted boundary method for embedded domain computations. Part II: Linear advection–diffusion and incompressible Navier–Stokes equations. *Journal of Computational Physics*, 372:996–1026, 2018.

[7] Helio JC Barbosa and Thomas JR Hughes. The finite element method with Lagrange multipliers on the boundary: circumventing the Babuška-Brezzi condition. *Computer Methods in Applied Mechanics and Engineering*, 85(1):109–128, 1991.

[8] Charles S Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3):220–252, 1977.

[9] Charles S Peskin. The immersed boundary method. *Acta numerica*, 11:479–517, 2002.

[10] Elie Hachem, Stephanie Feghali, Ramon Codina, and Thierry Coupez. Immersed stress method for fluid–structure interaction using anisotropic mesh adaptation. *International Journal for Numerical Methods in Engineering*, 94(9):805–825, 2013.

[11] Elie Hachem, Stephanie Feghali, Ramon Codina, and Thierry Coupez. Anisotropic adaptive meshing and monolithic Variational Multiscale method for fluid–structure interaction. *Computers & Structures*, 122:88–100, 2013.

[12] James A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[13] Paul Louis George, Houman Borouchaki, Frederic Alauzet, Patrick Laug, Adrien Loseille, and Loic Marechal. *Meshing, Geometric Modeling and Numerical Simulation, Volume 2: Metrics, Meshes and Mesh Adaptation.* John Wiley & Sons, 2019.

[14] Pascal-Jean Frey and Fr´ed´eric Alauzet. Anisotropic mesh adaptation for cfd computations. *Computer methods in applied mechanics and engineering*, 194(48-49):5068–5082, 2005.

[15] Frédéric Alauzet. Efficient moving mesh technique using generalized swapping. *In Proceedings of the 21st International Meshing Roundtable*, pages 17–37. Springer, 2013.

[16] Alexandre Ern and Jean-Luc Guermond. *Theory and practice of finite elements*, volume 159. Springer Science & Business Media, 2013.

[17] Thomas JR Hughes. Multiscale phenomena: Green's functions, the dirichlet-to-neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Computer Methods in Applied Mechanics and Engineering*, 127(1-4):387–401, 1995.

[18] Elie Hachem, Benjamin Rivaux, Thibaud Kloczko, Hugues Digonnet, and Thierry Coupez. Stabilized finite element method for incompressible flows with high reynolds number. *Journal of Computational Physics*, 229(23):8643–8665, 2010.

[19] B Gera and Pavan K Sharma. Cfd analysis of 2d unsteady flow around a square cylinder. International Journal of Applied Engineering Research Dindigul, 2010.

[20] Julien Bruchon, Hugues Digonnet, and Thierry Coupez. Using a signed distance function for the simulation of metal forming processes: Formulation of the contact condition and mesh adaptation. from a lagrangian approach to an eulerian approach. *International Journal for Numerical Methods in Engineering*, 78(8):980–1008, 2009.

[21] Kevin Wang, J Grétarsson, A Main, and C Farhat. Computational algorithms for tracking dynamic fluid–structure interfaces in embedded boundary methods. *International Journal for Numerical Methods in Fluids*, 70(4):515–535, 2012.

[22] Kevin Wang, Arthur Rallu, J-F Gerbeau, and Charbel Farhat. Algorithms for interface treatment and load computation in embedded boundary methods for fluid and fluid–structure interaction problems. *International Journal for Numerical Methods in Fluids*, 67(9):1175–1206, 2011.

[23] Paul E DesJardin, Brian T Bojko, and Matthew T McGurn. Initialization of high-order accuracy immersed interface cfd solvers using complex cad geometry. *International Journal for Numerical Methods in Engineering*, 109(4):487–513, 2017.

[24] Ramon Codina. Stabilization of incompressibility and convection through orthogonal sub-scales in finite element methods. *Computer Methods in Applied Mechanics and Engineering*, 190(13-14):1579–1599, 2000.

[25] Miguel Cervera, M Chiumenti, and Ramon Codina. Mixed stabilized finite element methods in nonlinear solid mechanics: Part i: Formulation. *Computer Methods in Applied Mechanics and Engineering*, 199(37-40):2559–2570, 2010.

[26] F. Brezzi, L.P. Franca, T.J.R. Hughes, and A. Russo. b $= \propto$ g. *Computer Methods in Applied Mechanics and Engineering*, 145(3):329–339, 1997.

[27] Thierry Coupez and Elie Hachem. Solution of high-reynolds incompressible flow with stabilized finite element and adaptive anisotropic meshing. *Computer Methods in Applied Mechanics and Engineering*, 267:65–85, 2013.

[28] Anita Hansbo and Peter Hansbo. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. *Computer Methods in Applied Mechanics and Engineering*, 191(47-48):5537–5552, 2002.

[29] Daniele Boffi and Lucia Gastaldi. A finite element approach for the immersed boundary method. *Computers & Structures*, 81(8-11):491–501, 2003.

[30] John Dolbow and Isaac Harari. An efficient finite element method for embedded interface problems. *International Journal for Numerical Methods in Engineering*, 78(2):229–252, 2009.

[31] Houman Borouchaki, Paul Louis George, and Bijan Mohammadi. Delaunay mesh generation governed by metric specifications part ii. applications. *Finite Elements in Analysis and Design*, 25(1):85–109, 1997. Adaptive Meshing, Part 1.

[32] Houman Borouchaki, Paul Louis George, Frédéric Hecht, Patrick Laug, and Eric Saltel. Delaunay mesh generation governed by metric specifications. part i. algorithms. *Finite Elements in Analysis and Design*, 25(1-2):61–83, 1997.

[33] Cyril Gruau and Thierry Coupez. 3d tetrahedral, unstructured and anisotropic mesh generation with adaptation to natural and multidomain metric. *Computer Methods in Applied Mechanics and Engineering*, 194(48-49):4951–4976, 2005.

[34] Ted Belytschko, Chandu Parimi, Nicolas Moës, Natarajan Sukumar, and Shuji Usui. Structured extended finite element methods for solids defined by implicit surfaces. *International Journal for Numerical Methods in Engineering*, 56(4):609–635, 2003.

[35] John Dolbow, NicolasMoës, and Ted Belytschko. An extended finite element method for modeling crack growth with frictional contact. *Computer Methods in Applied Mechanics and Engineering*, 190(51-52):6825–6846, 2001.

[36] Renato N Elias, Marcos AD Martins, and Alvaro LGA Coutinho. Simple finite element-based computation of distance functions in unstructured grids. *International Journal for Numerical Methods in Engineering*, 72(9):1095–1110, 2007.

[37] Guillermo Vigueras, Federico Sket, Cristobal Samaniego, Ling Wu, Ludovic Noels, Denny Tjahjanto, Eva Casoni, Guillaume Houzeaux, Ahmed Makradi, Jon M Molina-Aldareguia, et al. An xfem/czm implementation for massively parallel simulations of composites fracture. *Composite Structures*, 125:542–557, 2015.

[38] Antoine Legay, Jack Chessa, and Ted Belytschko. An eulerian–lagrangian method for fluid–structure interaction based on level sets. *Computer Methods in Applied Mechanics and Engineering*, 195(17-18):2070–2087, 2006.

[39] Qiming Zhu and Jinhui Yan. A mixed interface-capturing/interface-tracking formulation for thermal multi-phase flows with emphasis on metal additive manufacturing processes. *Computer Methods in Applied Mechanics and Engineering*, 383:113910, 2021.

[40] Facundo Del Pin, Sergio Idelsohn, Eugenio Oñate, and Romain Aubry. The The ALE/Lagrangian particle finite element method: a new approach to computation of free-surface flows and fluid–object interactions. *Computers & Fluids*, 36(1):27–38, 2007.

[41] Ruben Sevilla, Sonia Fernández-Méndez, and Antonio Huerta. Nurbs-enhanced finite element method(nefem). *Archives of Computational Methods in Engineering*, 18(4):441, 2011.

[42] Ramsharan Rangarajan and Adriá J Lew. Universal meshes: A method for triangulating planar curved domains immersed in nonconforming meshes. *International Journal for Numerical Methods in Engineering*, 98(4):236–264, 2014.

[43] HS Udaykumar, R Mittal, P Rampunggoon, and A Khanna. A sharp interface cartesian grid method for simulating flows with complex moving boundaries. *Journal of computational physics*, 174(1):345–380, 2001.

[44] Stanley Osher, Ronald Fedkiw, and K Piechor. Level set methods and dynamic implicit surfaces. *Appl. Mech. Rev.*, 57(3):B15–B15, 2004.

[45] James Albert Sethian. Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science, volume 3. *Cambridge university press*, 1999.

[46] Delphine Lacanette, Stéphane Vincent, Arthur Sarthou, Philippe Malaurent, and Jean-Paul Caltagirone. An eulerian/lagrangian method for the numerical simulation of incompressible convection flows interacting with complex obstacles: Application to the natural convection in the lascaux cave. *International Journal of Heat and Mass Transfer*, 52(11-12):2528–2542, 2009.

[47] C Farhat, A Rallu, K Wang, and T Belytschko. Robust and provably second-order explicit–explicit and implicit–explicit staggered time-integrators for highly non-linear compressible fluid–structure interaction problems. *International Journal for Numerical Methods in Engineering*, 84(1):73–107, 2010.

[48] Charbel Farhat, Arthur Rallu, and Sriram Shankaran. A higher-order generalized ghost fluid method for the poor for the three-dimensional two-phase flow computation of underwater implosions. *Journal of Computational Physics*, 227(16):7674–7700, 2008.