Transient Mesh Adaptivity with Large Rigid-Body Displacements

G. Compère*, J.-F. Remacle, and E. Marchandise

Institute of Mechanical, Materials and Civil Engineering (iMMC)
Université catholique de Louvain (UCL)
Avenue Georges Lemaitre 4, 1348 Louvain-la-Neuve, Belgium
{gaetan.compere,jean-francois.remacle,emilie.marchandise}@uclouvain.be

Summary. This paper presents a procedure for computing fluid-structure interaction problems when the boundaries of the domain undergo large displacements. The algorithm is based on both mesh motion and mesh adaptation techniques. More specifically, we use a node repositioning algorithm based on an elastic analogy together with a mesh adaptation procedure based on local mesh modifications [1]. This paper also includes a new technique for eliminating sliver tetrahedra. Some computational results are finally presented that include statistics on mesh quality measures.

1 Introduction

This paper deals with the issue of computing fluid-structure interaction (FSI) problems with large displacements of the structure. The most common way of dealing with FSI is to adopt an Arbitrary Lagrangian Eulerian (ALE) formulation of the fluid equations. ALE formulations allow to take into account small motions of the nodes in the fluid caused by the displacement of the structure. This approach suffers from obvious limitations as node repositioning cannot always provide a valid mesh when significant displacements or deformations of the structure are considered.

One way of addressing this problem is to remesh the entire domain when the displacements of the structure are too large to be handled [2, 3]. In this work, we rather use local mesh modifications [1, 4, 5, 6] both to optimize the quality of the tetrahedra and to comply a mesh size field. This approach is globally advantageous compared to global remeshing:

- Local solution projection procedures can be built in a way that ensures local conservation [4],
- The mesh remains unchanged in large parts of the domain,
- Local mesh modifications can be performed in parallel, enabling transient adaptive simulation to run on parallel computers [5].

There exist a small number of other approaches to handle large mesh deformations, among which the sliding mesh techniques [7]. Compared to local mesh

* Fonds de la Recherche Scientifique (FRS-FNRS), Belgium.

modifications, they have the disadvantage of requiring a flux computation at the interface and working with non coinciding meshes. The overset grid methods [8] are also developed in the literature but require grid assemblies and donor cell search processes. The local mesh modification technique has a unique mesh and a single prescribed boundary motion, which leads to a conceptually simple method and robust mesh movements [5].

Transient adaptive computations using local mesh modifications have already been applied to transient multiphase flow simulations in [9]. In this previous work, the mesh was adapted in order to capture the interface between two fluids, described by a level set function. In this paper, we extend the method to FSI problems.

In this context, two new issues have to be addressed. The first issue is the formulation of the mesh motion problem. In FSI, only the motion of the boundaries of the domain is prescribed. Typically, some kind of elastic analogy, possibly with a variable stiffness [10], is used to extend this motion inside the domain. Here, we do not make the assumption that the boundary motion is small or that the volume of the domain remains unchanged. An elastic approach does not give any guarantee on the validity of the mesh for large displacements. Even when the motion is limited, ill-shaped elements are produced in the process. This is the second issue. A new efficient procedure is presented that enables to eliminate all ill-shaped elements that are inevitably produced during the mesh motion.

The remainder of this paper is organized as follows: the first section introduces the different edge size fields. Section 3 describes the set of local mesh modification operators while section 4 presents the ill shaped elements elimination algorithm. In section 5, we recall the principles of the elastic analogy for mesh motion and discuss the choice of local element stiffness. The global procedure is then described in section 6. Section 7 presents some numerical results.

2 Mesh Size Field

The aim of the mesh generation process is to build elements of controlled shape and size. Mesh generators are usually able to adapt to a so called mesh size field (see for instance [11, 1]). An isotropic mesh size field is a scalar function $\delta(\mathbf{x}, t)$ that defines the optimal length of an edge at position $\mathbf{x}$ of the domain and at time t.

We typically define the non-dimensional length L_e^{tr} of edge e as

$$L_e^{tr}(t) = \int_e \delta^{-1}(\mathbf{x}, t) dl. \tag{1}$$

The quantity L_e^{tr} represents the number of subdivisions of edge e that are necessary for having an edge that has exactly the right size with respect to the size field.

There are lots of ways to define a size field: some are based on rigorous error estimation procedures [12] but there are lots of heuristics. For example, it is often considered that mesh size should be smaller near boundaries: when dealing with

viscous flows, a large part of the vorticity is created near the walls, when dealing with solid mechanics problems, stress concentration are usually located near the boundaries.

We define $d(\mathbf{x}, t)$ as the distance to the closest boundary at time t. This distance can be computed *in place* using the Approximated Nearest Neighbor Algorithm [13]. A first mesh size field $\delta_1(\mathbf{x}, t)$ is computed as follows:

$$\delta_1(\mathbf{x}, t) = \delta_1^{\text{small}} + \alpha_1(\mathbf{x}, t)(\delta_1^{\text{large}} - \delta_1^{\text{small}}),$$

where

$$\alpha_1(\mathbf{x}, t) = \begin{cases} 0 & \text{if} \quad d(\mathbf{x}, t) \leq d_1^{\min} \\ \frac{d(\mathbf{x},t) - d_1^{\min}}{d_1^{\max} - d_1^{\min}} & \text{if} \quad d_1^{\min} < d(\mathbf{x}, t) < d_1^{\max} \\ \infty & \text{if} \quad d(\mathbf{x}, t) \geq d_1^{\max} \end{cases}$$

with δ_1^{large} and δ_1^{small} a large and a small desired mesh sizes, $d_1^{\max}$ and $d_1^{\min}$ two field values that define the zone of refinement. An example of use of δ_1 is presented on figure 1.

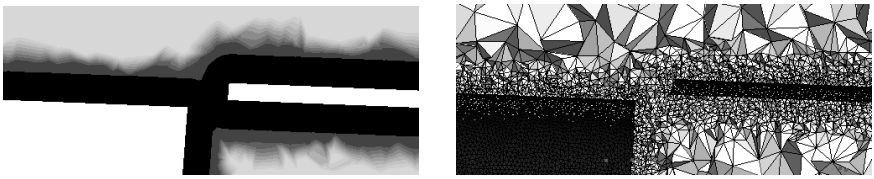


Fig. 1. Mesh adapted using the distance to boundaries

Other size fields $\delta_2(\mathbf{x}, t), \delta_3(\mathbf{x}, t), \ldots$ can be defined:

1. mesh sizes prescribed at model vertices and interpolated linearly on model edges;
2. prescribed mesh gradings on model edges (geometrical progressions, ...);
3. mesh sizes defined on another mesh (a background mesh) of the domain;
4. mesh sizes that adapt to the principal curvature of model entities.

The size field $\delta(\mathbf{x}, t)$ is computed, at time t, as the minimum of all size fields. It is usually bounded by upper and lower values of mesh sizes.

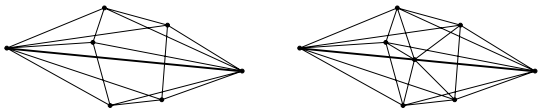
In an ideal mesh, each edge has an adimensional length $L_e^{tr} = 1$. This ideal situation cannot be attained in practice. In the adaptation procedure one has to decide whether an edge is acceptable, i.e. find a range $[L_{low}, L_{up}]$ for which an edge is considered to have a good size. Then a long edge has a size $L_e^{tr} > L_{up}$ and a short edge has a size $L_e^{tr} < L_{low}$.

In [1], the authors show that choosing a too narrow range for acceptable edge sizes may lead to infinite loops between splits and collapses. In § 6 and § 7, we show the influence of this interval on the mesh quality and on the number of infinite loops.

3 Local Mesh Modifications

The three basic local mesh modifications are the edge split, the edge collapse and the edge swap. Other operators like face swap and some compound operators are also used in this work.

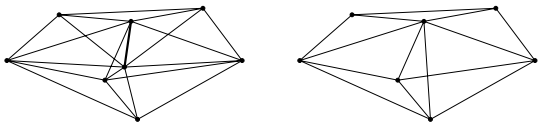
There are two ways of dealing with long edges. The first one consists in tagging every long edge of the mesh. Then, every tetrahedron of the mesh is subdivided using a template that is function of the number of edges that have been split. The second manner consists in splitting all tetrahedra surrounding a long edge, and then proceed to the next edge. The latter approach has been used because it is simpler (only one template has to be defined), more robust (no Steiner point has to be introduced) and (surprisingly) more efficient. Moreover, we have found out that both methods lead to meshes with elements of similar qualities. **The edge split** operator is depicted on figure 2.



(a) (b)

Fig. 2. Edge split operation: (a) initial cavity, (b) cavity after the edge split

When a short edge is found, an **edge collapse** is applied. The edge collapse operator (see figure 3) removes an edge and all its bounding elements from the mesh by merging its two extremities at one of their locations.



(a) (b)

Fig. 3. Edge collapse operation: (a) initial cavity, (b) cavity after the edge collapse

The edge swap consists in remeshing the cavity surrounding the edge with the aim of improving the worst element shape in that cavity. Figure 4(a) shows an edge to be swapped that is surrounded by five tetrahedra. Figures 4(c), (d) and (e) depict three of the five possible configurations after the swap. The retained configuration is the one that has the best minimal element quality in the cavity.

The face swap removes a face and replaces it by an edge, leading to the creation of three tetrahedra instead of two. This operation can be seen as the inverse edge swap with $n = 3$.

The face collapse operator can be seen as the compound of and edge split and an edge collapse. The operation is depicted on Figure 5. One of the edges of

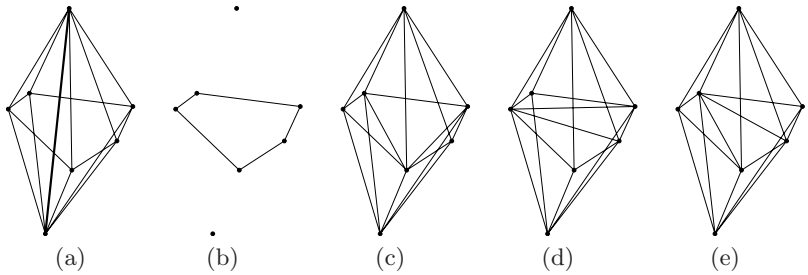


Fig. 4. Edge swap operation: (a) initial cavity with $n = 5$, (b) mean surface that is going to be triangulated, (c),(d) and (e) three possible configurations after the swap

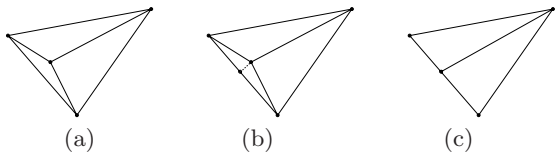


Fig. 5. Face collapse operation: (a) initial tetrahedron, (b) split of an edge of the concerned face, (c) collapse of the new edge on the new vertex

the face is split and the resulting new edge is collapsed on the new vertex. This operation is particularly well suited to eliminate an ill-shaped element that has a face with a small ratio area/edge lengths.

The double edge split collapse compound operator consists in splitting two edges of a tetrahedron and then collapsing the edge joining the two new vertices. This operator is used when a tetrahedron with a small volume has two edges nearly intersecting. The operation is depicted on Figure 6.

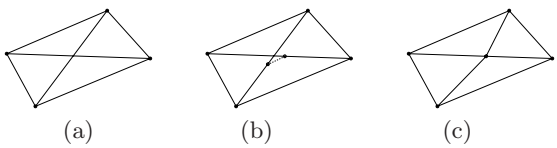


Fig. 6. Double edge split + edge collapse operation: (a) initial tetrahedron, (b) situation after the edge splits, (c) situation after the edge collapse

4 Sliver Tetrahedra Handling

A tetrahedron is said to be a sliver when it has a small volume and no short edge. Such a tetrahedron can be classified in one of the two categories [14] depicted

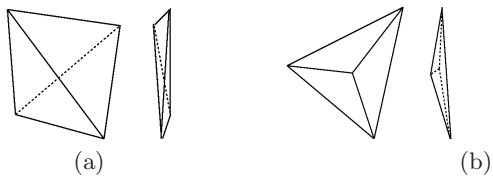


Fig. 7. Classification of the sliver tetrahedra: (a) type I, (b) type II

on figure 7. Type I slivers are tetrahedra in which two edges almost intersect. In type II slivers, one vertex is very close to its opposite face. The determination of the sliver type is important because it gives a useful information about the best local mesh modification that can be applied in order to eliminate it. The projection algorithm presented in [1] is used in this work to determine the type of sliver.

In FSI, when the mesh motion solver is pushed to its limits, it inevitably generates slivers near the boundaries undergoing a motion. In [1], an algorithm selecting the local mesh modifications to be applied was proposed to eliminate most of them. In this work, we propose an extension to this algorithm that eliminates more slivers.

A new sequence of mesh modification operations is proposed in Table 1. The operations are sorted with respect to their average efficiency to eliminate slivers of type I and II. The efficiency measure that we have used takes into account both the rate of success of eliminating slivers and CPU time consumption of the given operator. The rates of success of the different operations have been deduced from various test cases. Some of them are presented in section 7. We notice that the vertex motion is the very last solution as it is highly probable that the tetrahedron becomes a sliver again after the next step of the boundaries motion. The target location for the motion is computed with the method proposed in [15].

Note that those sequences are designed to be part of the sliver elimination algorithm inserted in the global adaptation procedure presented in section 6. Other sequences could be more efficient if the global procedure was modified.

In this work, we consider that eliminating the slivers is more important than enforcing locally the length criterion, which means that an operation that can eliminate a sliver or at least improve the local quality is performed, even if it creates long or short edges. The long or short edges will be eliminated later on by the global procedure but only if they do not create a sliver tetrahedron (see section 6.2).

The construction of the list of sliver tetrahedra is made by computing the quality of every element of the mesh. The quality measure is chosen to be the cubic mean ratio [16], defined by

$$\eta^3 = 15552 \frac{V^2}{\left(\sum_{i=1}^{6} (L_i)^2\right)^3}, \tag{2}$$

Table 1. Sequence of local mesh modifications attempted to eliminate a sliver

Sliver type	Priority level	Local mesh modification
I	1	Split one of the two key edges
	2	Collapse one of the edges of the tetrahedron
	3	Split both key edges and collapse the new interior edge
	4	Split one of the key edges and collapse the new vertex
	5	Swap one of the two key edges
	6	Relocate a vertex
II	1	Collapse one of the edges of the tetrahedron
	2	Collapse the key face
	3	Swap any of the edges bounding the key face
	4	Swap the key face
	5	Relocate a vertex

with V the volume of the element and L_i the length of the ith edge of the element. The 15552 factor is set to scale η^3 such that it ranges from 0 to 1. Its value is 1 for an equilateral tetrahedron and 0 for a flat tetrahedron. An element is considered as a sliver if its quality is under a certain threshold T_{sliver}. In all simulations presented in this paper, T_{sliver} is set to 0.01.

5 Mesh Motion Solver

In FSI problems, the structure imposes its motion to some of the boundaries of the fluid domain. Moving only the boundary nodes leads to the generation of ill conditioned tetrahedra. A way to circumvent partially this issue is to relocate the nodes of the volume using a linear elasticity analogy [17, 6].

A linear elastic problem is solved every time the mesh is moved. Here, we use the approach of [10] in which a stiffness alteration based on the Jacobian of the elements is used to stiffen the smaller elements. The corresponding finite element formulation is written as follows. Consider the vector of displacements of the nodes $\mathbf{y}^h \in \mathcal{S}^h$, where $\mathcal{S}^h$ is the piecewise linear nodal finite element space. The modified energy of deformation of a tetrahedron e submitted to a displacement $\mathbf{y}^h$ (with the corresponding strain and stress tensors $\boldsymbol{\epsilon}$ and $\boldsymbol{\sigma}$) is written as:

$$\frac{1}{2}\int_e \left[\boldsymbol{\epsilon}(\mathbf{y}^h) : \boldsymbol{\sigma}(\mathbf{y}^h)\right]^e J^e \left(\frac{J^0}{J^e}\right)^{\chi} dv, \tag{3}$$

where χ is the stiffening parameter and J^0 a fictious volume constant over the mesh.

Figure 8 illustrates the influence of χ. With $\chi = 0$, we observe ill shaped elements near the walls. The elastic solver fails to achieve the mesh deformation with positive volumes of the elements if a motion of the cube of more than $0.1 \times h$ where h is the size of the cube is imposed. With $\chi = 1$ and $\chi = 2$, a motion of $1.0 \times h$ can be attained without any particular problem. In the $\chi = 2$ case, the

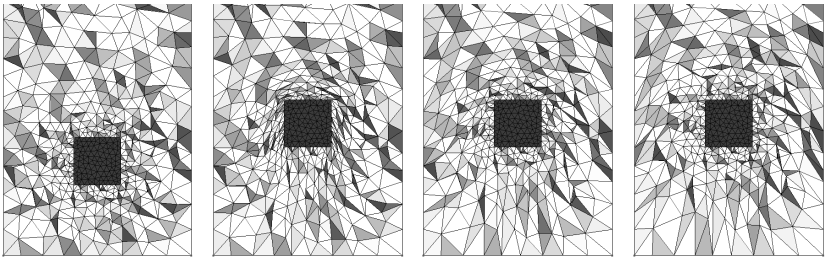


Fig. 8. Mesh around a cube and its deformations for various values of χ

elements are very well preserved around the cube but the quality falls outside of the refined zone. The choice of $\chi = 1$ seems to be a good compromise and has been used in all the computations presented in this paper.

6 Global Mesh Modification Procedure

Our FSI procedure has two different stages: the node repositioning stage and the mesh adaptation stage. The node repositioning step does not imply any change in the mesh topology. As it does not need to reallocate the resources for the storage of the topology, the solution or any data, it can be performed often at a reasonable computational cost, that is, each time a displacement is prescribed at the interface between the fluid and the structure. Moreover, the node repositioning is explicitly taken into account in the ALE formulation of the governing equations and does not need any mesh to mesh interpolation. In the second step of the procedure, the mesh is adapted using the local mesh modifications presented in section 3 and the global size field presented in section 2.

6.1 The Mesh Adaptation Algorithm

The aim of the adaptation procedure is twofold. First, the adimensional size of all edges of the mesh have to lie in the interval $[L_{low}, L_{up}]$. Then, the quality of the resulting mesh has to be optimal. Minimal quality criterions can be expressed for instance in terms of a minimal quality for every element and/or a minimal mean quality of the elements.

The mesh adaptation procedure can be described as follows:

```
Do {
```

- Collapse short edges i.e. edges that have lengths $L_e^{tr} < L_{low}$.
- Do an edge swap loop: loop over all edges and compute the minimal quality of the elements surrounding the edge. The quality is given by the cubic mean ratio (equation 2). If the quality is lower than a given threshold T_{swap}, try to find a swap configuration. Apply the swap if it improves the minimal quality

in the cavity and if it does not create a long or a short edge. The threshold T_{swap} is fixed to 0.1 in all the simulations presented in this paper.

- Eliminate sliver tetrahedra with the algorithm described in section 4.
- Split long edges i.e. edges that have lengths $L_e^{tr} > L_{up}$.

} `While the mesh is modified.`

The collapse loop is performed first in order to avoid memory peaks. Indeed, the number of nodes decreases during this loop. As the edge swap loop is costly, it is performed just after the coarsening loop, with the minimum number of nodes. The sliver region elimination can then be performed without having to compute any element shape, as all element qualities are computed at the previous step.

6.2 Infinite Loops

There are a few ways in which the adaptation procedure can degenerate in an infinite loop between two or more configurations.

If the interval of tolerance for the length of the edges is not large enough, an infinite loop between split and collapse operations can appear. Due to the heuristic nature of the mesh adaptation, we cannot guarantee that such a loop will never appear, even for a large interval. For that reason, a maximum number of iterations is imposed in the global procedure. The edge length interval should then be carefully chosen in order to strongly limit the number of infinite loops without being too far from the unit mesh. More detailed results are presented in section 7.

Another possibility of having an infinite loop is illustrated in Figure 9, in which an edge split is followed by an edge collapse and an edge swap. This is possible because the goal of the edge split and collapse operators is to respect a criterion on edges length while the edge swaps tend to improve the quality of the tetrahedra. A way to eliminated such a scenario is to forbid an edge swap if it creates a long or short edge.

Finally, the operators used in the sliver regions handler can also create a set of complex infinite loops. This possibility can be avoided if any operation that creates a sliver tetrahedron is forbidden. Of course, as an exception, we allow the replacement of a sliver by a better sliver.

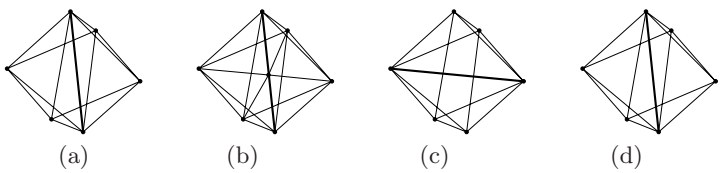


Fig. 9. Infinite loop between an edge swap, an edge split and an edge collapse: (a) initial cavity, (b) cavity after the split, (c) cavity after the collapse, (d) cavity after the swap, identical to (a)

7 Computational Results

Our adaptation procedure has been evaluated on three test cases. The quality of the meshes, the elimination of the sliver tetrahedra and the evolution of the number of nodes are presented in the first and the second cases. The third test case is a fluid-structure interaction computation with a very large motion of one or two spheres in a fluid at rest.

7.1 Cylinder and Tube

This test case consists in the penetration of a cylinder in a tube, the diameters of the objects being quite close. The initial geometry and mesh are shown in figure 10. The cylinder has a radius of 0.9 and a length of 3.0. The tube has the same length with an internal radius of 1.0 and an external radius of 1.2. The initial distance between the two objects is 1.0. The two objects move with a velocity of 1.0 for the cylinder and -1.0 for the tube. The end of the test is fixed at time 4.0, when the objects are completely separated and the distance between them is 1.0.

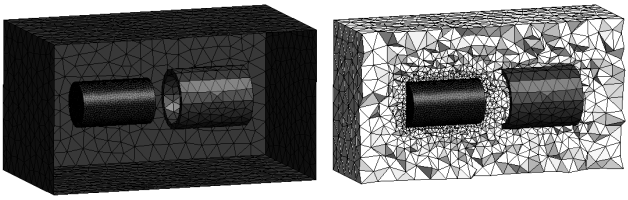


Fig. 10. Cylinder and tube test case: initial geometry and mesh

A global edge length of 0.6 is prescribed on the whole domain and both objects are equipped with a local size field. The parameters of the local size fields for the cylinder and the tube are

$$\begin{cases} d_{cyl}^{\min} = 0 \\ d_{cyl}^{\max} = 1.0 \end{cases} \qquad \begin{cases} \delta_{cyl}^{\text{small}} = 0.1 \\ \delta_{cyl}^{\text{large}} = 0.6 \end{cases} \qquad \begin{cases} d_{tube}^{\min} = 0 \\ d_{tube}^{\max} = 1.0 \end{cases} \qquad \begin{cases} \delta_{tube}^{\text{small}} = 0.4 \\ \delta_{tube}^{\text{large}} = 0.6 \end{cases}$$

For an interval $[L_{low}, L_{up}]$ set at $[0.5, 1.4]$, the number of nodes ranges from about 23.500 to 33.000 during the adaptation. This is sufficient to get representative data and statistics about the mesh quality and the sliver elements handling.

A time step of 0.01 has been chosen for this test case. It corresponds to a relative motion of the objects of $0.2 d_{cyl}^{\min}$ at each time step.

The mesh aspect obtained at different times with a time step of 0.01 and an interval $[L_{low}, L_{up}]$ set at $[0.5, 1.4]$ is shown in figure 11.

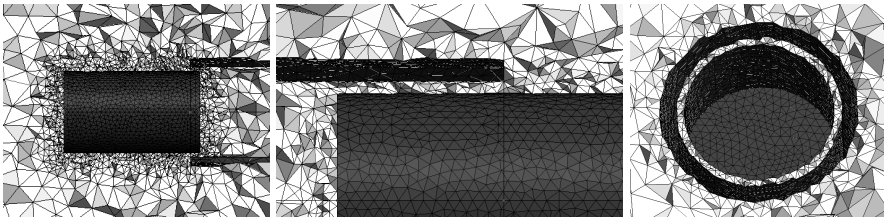


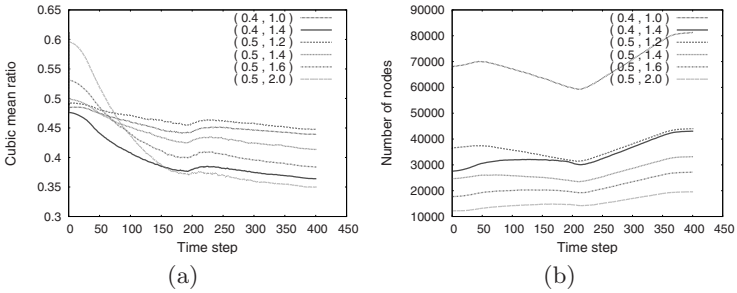
Fig. 11. Mesh aspect at different times with $\Delta t = 0.01$, $L_{low} = 0.5$ and $L_{up} = 1.4$

Table 2. Tube and cylinder test case: statistics about sliver tetrahedra handling

		Successes	New sliver	Failures	Success rate (%)
Type I (547)					
	All operators	541	1	5	98.9
	Edge collapse	244	3	300	44.6
	Edge split	346	20	181	63.3
	Double split+collapse	210	11	326	38.4
	Face collapse	279	13	255	51.0
	Edge swap	306	17	224	55.9
	Vertex relocation	504	10	33	92.1
Type II (308)					
	All operators	306	1	1	99.4
	Edge collapse	119	2	187	38.6
	Face collapse	155	1	152	50.3
	Edge swap	202	5	101	65.6
	Face swap	112	6	190	36.4
	Vertex motion	292	1	15	94.8

Sliver tetrahedra handling

In order to get more information about the efficiency of each mesh modification applied in the sliver region elimination, we have run the adaptation procedure in which the slivers handling algorithm is enhanced with a routine that tests all operations individually on each sliver before going in the normal algorithm of elimination. This routine has no influence on the results, but provides the statistics shown in table 2. The first columns indicates the number of sliver tetrahedra that the operators can eliminate without creating a new sliver. The second column shows the number of situations in which the operator can only eliminate the sliver by creating another sliver with a better quality, while the third column shows the number of slivers that could not be eliminated or for which a worse sliver would appear. The rate of success of the operator (not including the cases in which a new sliver is created) is indicated in the last column.



(a) (b)

Fig. 12. Cylinder and tube test case: comparison between intervals $[L_{low}, L_{up}]$: (a) evolution of the mean quality of the mesh and (b) the number of nodes

Table 3. Tube and cylinder test case: comparison between intervals $[L_{low}, L_{up}]$: number of infinite loops

Interval	$[0.4, 1.0]$	$[0.4, 1.4]$	$[0.5, 1.2]$	$[0.5, 1.4]$	$[0.5, 1.6]$	$[0.5, 2.0]$
Infinite loops	400	0	400	81	0	0

We can see that combining all operators do not lead to a total elimination of the slivers but provides a very good rate of elimination. The vertex repositioning looks attractive but we observed that a lot of slivers eliminated by a repositioning become slivers again once a global node motion is applied. For type I slivers, the edge split is very efficient if we notice that it is a very fast operator compared to the edge swaps. Note that those results are influenced by the fact that the sliver elimination is done after the edge collapse and edge swap loops and before the edge split loop.

Mesh quality and infinite loops

A particular result that we observe in our simulations is that the mean quality of the mesh depends on the boundaries of the interval $[L_{low}, L_{up}]$. Figure 12 (a) shows the evolution of the quality for various intervals, figure 12 (b) shows the evolution of the number of nodes in each case, while table 3 shows the number of time steps in which an infinite loop was created.

We can see that the condition $L_{low} < 0.5\, L_{up}$ is not sufficient to avoid infinite loops. Indeed, there is an infinite loop at each time step with the intervals $[0.4, 1.0]$ and $[0.5, 1.2]$ while some infinite loops appear with $[0.5, 1.4]$. We would then recommend to use an interval with a similar range than $[0.4, 1.4]$ or $[0.5, 1.6]$.

We also observe that the quality is dependent on the boundaries of the interval. If we compare the intervals $[0.4, 1.4]$ and $[0.5, 1.4]$, the quality seems to be better with the largest L_{low}. If we look at the results for $[0.5, L_{up}]$, we can see that the quality decreases to a smaller value when L_{up} is bigger. These two observations

lead to the conclusion that the quality tends to a bigger value if the interval is smaller.

In order to get the best mean quality without running into infinite loops, a compromise has to be found on the size of the interval. The smaller interval which do not lead to infinite loop should be used. From the set of intervals tested here above, our choice would turn towards $[0.4, 1.4]$ or $[0.5, 1.6]$.

7.2 Worm Screw

The second test case is a worm screw whirling inside a cylinder. It has been chosen to demonstrate the capabilities of the adaptation procedure on relatively complex geometries.

Figure 13 shows the geometry and the initial mesh of the domain at time 0. The domain is bounded by a worm screw of diameter 4.94 and length 15.6 turning with an angular speed of 1.0, and a fixed cylinder of diameter 5.2 and length 16.2. The computation ends when a complete turn has been operated, i.e. at $t = 2\pi$. A time step of 0.02 is chosen.

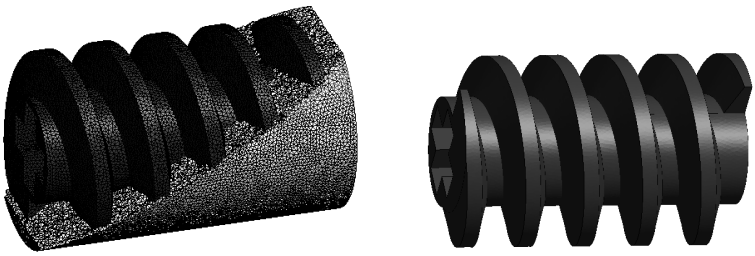


Fig. 13. Worm screw: initial geometry and mesh

The maximum size on the whole domain is set to 1.0, while a size field with the following parameters is prescribed at every wall:

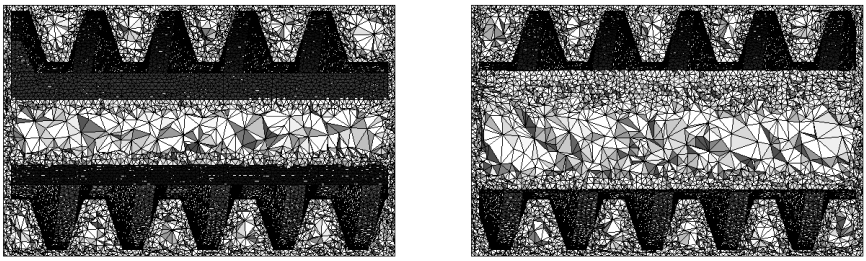
$$d_1^{\min} = 0 \qquad d_1^{\max} = 0.4 \qquad \delta_1^{\text{small}} = 0.2 \qquad \delta_1^{\text{large}} = 0.4$$

The interval $[L_{low}, L_{up}]$ is set to $[0.5, 1.6]$. With these parameters, the number of nodes ranges from 125.000 to 155.000 during the computation.

Figure 14 shows the aspect of the mesh at times 4.0 and 6.0.

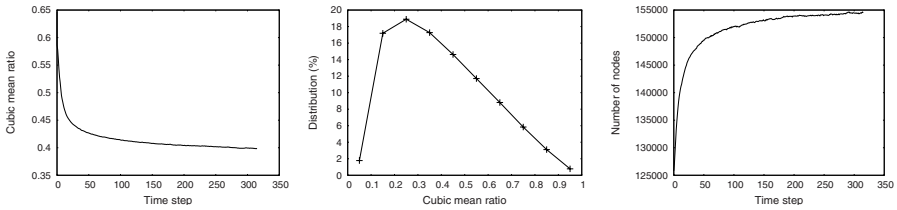
The evolution of the mean quality is shown in figure 15 (a). The cubic mean ratio tends to a value close to 0.40, which is quite similar to what was observed in the previous test case with the same interval $[L_{low}, L_{up}]$. The distribution of the quality in the final mesh is shown in figure 15 (b). The evolution of the number of nodes is drawn on figure 15 (c). It stabilizes around 155.000.

The statistics about the elimination of the sliver tetrahedra are shown in table 4. Every sliver is eliminated and most of them are eliminated at the first attempts.



(a) (b)

Fig. 14. Worm screw: cuts in the mesh at times 4.0 and 6.0



(a) (b) (c)

Fig. 15. Worm screw: (a) evolution of the mean cubic mean ratio, (b) distribution of the quality in the final mesh, (c) evolution of the number of nodes

Table 4. Worm screw: statistics about sliver tetrahedra handling

		Applied (no sliver)	Applied (new sliver)
Type I (1532)			
	Edge split	1182	0
	Edge collapse	239	0
	Double split+collapse	15	1
	Face collapse	18	0
	Edge swap	4	1
	Vertex relocation	72	0
	Remaining	0	
Type II (1521)			
	Edge collapse	1430	0
	Face collapse	33	1
	Edge swap	17	0
	Face swap	1	0
	Vertex motion	38	1
	Remaining	0	

7.3 Spheres Falling in a Fluid

In this test case, the mesh adaptation procedure is applied to a fluid-structure interaction solver. The problem solved implies the motion of one or two spheres falling in a fluid.

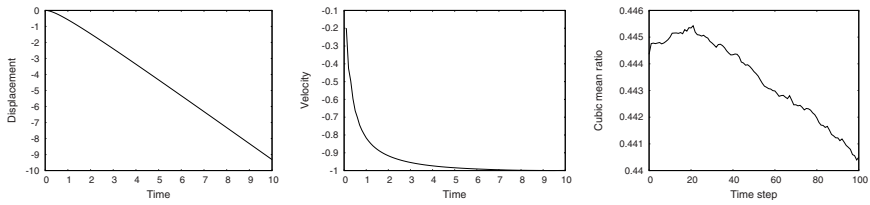
The fluid is governed by the incompressible Navier-Stokes equations. The equations are discretized in a dual finite elements / finite volumes formulation.

The motion of the sphere is computed with the Newmark method and the coupling is achieved with a CSS procedure [18] equipped with sub-iterations cycles.

Single sphere

For this computation, a single sphere is immersed in the fluid. We fix the parameters of the fluid so that the Reynolds number Re is equal to 1 at equilibrium. Due to the gravity, the sphere accelerates until the gravity is exactly balanced by the drag force and the Archimede's force. We fix the mass of the sphere such that the theoretical velocity at equilibrium is 1.

Figures 16 shows the evolution of the velocity and displacement of the sphere as well as the evolution of the mesh quality. We notice that the velocity tends to 1 as expected and that the quality decreases slowly so that the loss along the computation is not significant.



(a) (b) (c)

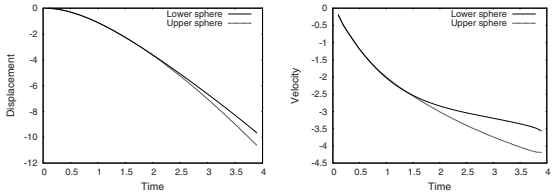
Fig. 16. Single sphere test case: (a) evolution of the displacement, (b) velocity and (c) mean quality of the mesh

Two spheres

In the next computation, the motion of two vertically aligned spheres are investigated.

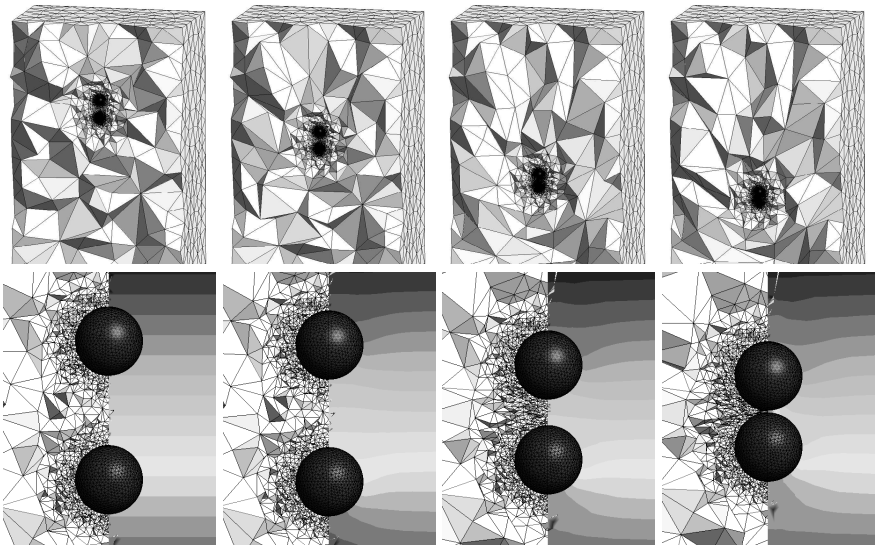
The evolution of the displacements and the velocities of the spheres are shown in figures 17 (a) and (b). We observe that from the time 1.5, the upper sphere is aspirated by the flow around the first one. Eventually, the spheres touch and the computation stops.

The cut of the computational mesh is represented on figure 18 for different times as well as the pressure field.



(a) (b)

Fig. 17. Two spheres test case: (a) evolution of the displacements and (b) velocities



Time: 0.0 2.0 3.4 3.9

Fig. 18. Two spheres test case: mesh and pressure field at different time steps

8 Conclusion

A procedure to handle large deformations of a mesh in FSI problems by nodes movement and local mesh adaptation was presented. The procedure is robust and can handle complex geometries with very large deformations of the domain.

For that purpose, a node repositioning technique with a selective treatment for the elements previously applied for two-dimensional meshes was applied for three-dimensional meshes.

A new adaptation procedure has been proposed in which an efficient handling of the sliver tetrahedra is included. The procedure achieves a good quality for the mesh and complies a mesh size field. Important parameters of the procedure like the boundaries of the interval $[L_{low}, L_{up}]$ were studied with a set of computations.

In particular, some clues are given to choose them by looking at the quality, number of nodes and production of infinite loops.

Finally, mesh deformation tests and fluid-structure computations have been performed in order to show the potential of the presented approach.

References

1. Li, X., Shephard, M.S., Beall, M.W.: 3d anisotropic mesh adaptation by mesh modification. Computer Methods in Applied Mechanics and Engineering 194, 4915–4950 (2005)
2. Almeida, R., Feijoo, P., Galeao, A., Padra, C., Silva, R.: Adaptive finite element computational fluid dynamics using an anisotropic error estimator. Computer Methods in Applied Mechanics and Engineering 182(3-4), 379–400 (2002)
3. Borouchaki, H., George, P., Hecht, F., Laug, P., Saltel,: Delaunay mesh generation governed by metric specifications - part i: Algorithms and part ii: Applications. Finite Elements in Analysis and Design 25, 61–83, 85–109 (1997)
4. Remacle, J.-F., Li, X., Shephard, M., Flaherty, J.: Anisotropic adaptive simulation of transient flows using Discontinuous Galerkin methods. International Journal for Numerical Methods in Engineering 62, 899–923 (2004)
5. Cavallo, P.A., Sinha, N., Feldman, G.M.: Parallel unstructured mesh adaptation method for moving body applications. AIAA Journal 43, 1937–1945 (2005)
6. Baker, T.J.: Mesh movement and metamorphosis. Engineering With Computers 18(3), 188–198 (2002)
7. Mathur, S.R.: Unsteady flow simulations using unstructured sliding meshes. AIAA Journal (June 1994)
8. Lijewski, L.E., Suhs, N.E.: Time-accurate computational fluid dynamics approach to transonic store separation trajectory prediction. Journal of Aircraft 31(4), 886–891 (1994)
9. Compère, G., Marchandise, E., Remacle, J.-F.: Transient adaptivity applied to free surface flows. Journal of Computational Physics 227, 1923–1942 (2008)
10. Stein, K., Tezduyar, T., Benney, R.: Mesh moving techniques for fluid-structure interactions with large displacements. Journal of Applied Mechanics 70, 58–63 (2003)
11. Peraire, J., Peiro, J., Morgan, K.: Adaptive remeshing for three dimensional compressible flow computation. Journal of Computational Physics 103, 269–285 (1992)
12. Ainsworth, M., Oden, J.T.: A Posteriori Error Estimation in Finite Element Analysis. Wiley, New York (2000)
13. Arya, S., Mount, D.M., Netanyahu, N.S., Silverman, R., Wu, A.Y.: An optimal algorithm for approximate nearest neighbor searching. Journal of the ACM 45, 891–923 (1998), `http://www.cs.umd.edu/~mount/ANN/`
14. de l'Isle, E.B., George, P.L.: Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations, I. In: Babuska, I., et al. (eds.), vol. 75, pp. 97–128. Springer, Heidelberg (1993)
15. de Cougny, H.L., Shephard, M.S., Georges, M.K.: Explicit node point mesh smoothing within the octree mesh generator. Scorec report, Rensselaer Polytechnic Institute, 10 (1990)
16. Liu, A., Joe, B.: On the shape of tetrahedra from bisection. Mathematics of computation 63, 141–154 (1994)

17. Tezduyar, T.E., Behr, M., Mittal, S., Liou, J.: Computation of unsteady incompressible flows with the finite element methods–space-time formulations, iterative strategies and massively parallel implementations vol. 246, ASME Pressure Vessels Piping Div Publ PVP., ASME, NY, pp. 7–24 (1992)
18. Piperno, S., Farhat, C.: Partitioned procedures for the transient solution of coupled aeroelastic problems - part ii: energy transfer analysis and three-dimensional applications. Computer Methods in Applied Mechanics and Engineering 190, 3147–3170 (2001)