Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers and Structures 83 (2005) 155–165
Computers & Structures
www.elsevier.com/locate/compstruc

Fluid–structure interaction in blood flows on geometries based on medical imaging

Jean-Frédéric Gerbeau [a,*], Marina Vidrascu [b], Pascal Frey [c]

[a] Projet REO, INRIA Rocquencourt, B.P. 105, 78153 Le Chesnay Cedex, France
[b] Projet MACS, INRIA Rocquencourt, B.P. 105, 78153 Le Chesnay Cedex, France
[c] Laboratoire Jacques-LouisLions, Universite Pierre et Marie Curie, Boite Courier 187, 75252 Paris Cedex 5, France

Accepted 5 March 2004

Abstract

We address two difficult points in the simulation of blood flows in compliant vessels: the fluid and structure meshes generation and the solution of the fluid–structure problem with large displacements. The proposed strategy allows to perform realistic simulations on geometries coming from medical imaging.

Keywords: Fluid–structure interaction; Mesh generation; Newton–Krylov algorithms; Added-mass effect; Blood flows

1. Introduction

The main objective of this article is to present a feasible strategy to simulate blood flows in compliant large vessels. Such simulations raise many difficulties. We specifically address two of them: first, the mesh generation for the fluid and the wall and second, the efficient and robust resolution of the nonlinear fluid–structure problem.

A structural model for large arteries has to be three-dimensional and has to handle large displacements. Given that the wall of the blood vessels is thin, it is convenient to use shell elements. For ''realistic'' simulations, a procedure that starts with medical images and automatically generates a finite element mesh has to be designed. The most efficient and reliable procedures able to achieve this goal produce tetrahedral meshes and thus triangular meshes of the arteries surfaces. In this context, it could be appropriate to use triangular shell elements. Nevertheless, it is usually admitted that quadrilateral shell elements are more reliable in general situations. We therefore have to couple quadrilateral meshes for the structure and tetrahedral meshes for the fluid. A possible strategy could be to use a method—for example interpolation or mortar element [5]—to manage non-matching grids. But the practical implementation of such techniques on general 3D geometries is rather involved. In the present work, we propose an alternative which consists in converting pairs of triangles of the tetrahedral mesh skin in order to produce a quadrilateral mesh. Of course, this procedure has to be carried out very carefully to ensure the quality of the resulting mesh. By the way, the method proposed in this paper is a general way to generate a non-structured quadrilateral mesh of a surface. As a result of this approach, the vertices of the quadrilateral and the

* Corresponding author. Tel.: +33 1 39 63 57 48; fax: +33 1 39 63 58 82.
E-mail address: jean-frederic.gerbeau@inria.fr (J.-F. Gerbeau).

0045-7949/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compstruc.2004.03.083

tetrahedral meshes can be matched, which makes the message passing between the fluid and structure solvers very simple.

Once suitable meshes have been generated, the fluid–structure problem has to be tackled. To this purpose, partitioned schemes are very convenient: they allow to use the available software with minor changes, the most adapted schemes for both fluid and structure solvers, and they ensure that the fluid–structure solver will automatically inherit future improvements in fluid or structure algorithms. For a presentation of partionned and simultaneous procedures for fluid–structure problems we refer to [36,40,2,3]. In the family of partitioned schemes, loosely-coupled algorithms (see [15] and the references therein) are very attractive since they typically require one (or a few) resolution(s) of fluid and structure at each time step. Such algorithms are very efficient in aeroelasticity [15,26,34], but it has been observed in many works [12,23,33] that strongly-coupled algorithms seem to be mandatory in blood flows. Partitioned strongly-coupled methods yield the resolution of a nonlinear problem on the fluid–structure interface [25], which may be very time-consuming. In particular, classical fixed-point methods are too expensive, and sometimes not robust enough (even if acceleration techniques improve their efficiency [29–31]). This fact, which is now well-understood [10,25], is mainly due to the *added-mass effect* (see for example [32] for a presentation of this effect in various applications). We present in this article a Jacobian-free Newton–Krylov method based on the following basic idea: a simplified problem which takes into account the added-mass effect can be used to efficiently evaluate an approximate product "Jacobian times vector" in a Krylov method. The resulting algorithm appears to be efficient and robust in the context of blood flows.

The article is organized as follows. We present in Section 2 the basic equations of the problem. Section 3 is devoted to the description of the mesh generation algorithms. In Section 4, the shell model, the load computation and the time marching algorithms are detailed. The inexact Newton method is described in Section 5 and some numerical results are presented in Section 6.

2. General formulation of the fluid–structure problem

Let $\Omega(t)$ be a time-dependent domain of $\mathbb{R}^3$ occupied by a continuum medium. It is assumed that, for all time t, $\overline{\Omega(t)} = \overline{\Omega_{\mathrm{F}}(t)} \cup \overline{\Omega_{\mathrm{S}}(t)}$ and $\Omega_{\mathrm{F}}(t) \cap \Omega_{\mathrm{S}}(t) = \emptyset$, where $\Omega_{\mathrm{F}}(t)$ is occupied by a fluid and $\Omega_{\mathrm{S}}(t)$ is occupied by an elastic solid. We denote by $\Sigma(t) = \overline{\Omega_{\mathrm{F}}(t)} \cap \overline{\Omega_{\mathrm{S}}(t)}$ the fluid–structure interface. The domain $\Omega(t)$ is the current configuration of the system. Let $\hat{\Omega}$ be a reference configuration. We define the deformation $\hat{\boldsymbol{\varphi}}$ of the continuum medium:

$$\hat{\boldsymbol{\varphi}} : \hat{\Omega} \times [0,T] \to \Omega(t)$$

$$(\hat{x},t) \to x = \hat{\boldsymbol{\varphi}}(\hat{x},t),$$

the deformation gradient:

$$\hat{\boldsymbol{F}}(\hat{x},t) = \nabla_{\hat{x}} \hat{\boldsymbol{\varphi}}(\hat{x},t),$$

and its determinant $\hat{J}(\hat{x},t) = \det \hat{\boldsymbol{F}}(\hat{x},t)$. The curve $t \to \hat{\boldsymbol{\varphi}}(\hat{x},t)$ corresponds to the trajectory of the material particle $\hat{x}$. We define the displacement

$$\hat{\boldsymbol{d}}(\hat{x},t) = \hat{\boldsymbol{\varphi}}(\hat{x},t) - \hat{x}.$$

The velocity of a particule $\hat{x}$, defined as $\frac{\partial \hat{\boldsymbol{\varphi}}}{\partial t}(\hat{x},t)$, will be denoted $\hat{\boldsymbol{u}}_{\mathrm{s}}$ in the structure and $\hat{\boldsymbol{u}}$ in the fluid. On a point x in the fluid, $\boldsymbol{u}(x,t)$ denotes the quantity $\hat{\boldsymbol{u}}(\hat{x},t)$, where $x = \hat{\boldsymbol{\varphi}}(\hat{x},t)$.

Following the Arbitrary Lagrangian Eulerian (ALE) approach, we introduce another mapping which does not coincide in general with the material trajectories. We define the fluid domain deformation:

$$\hat{\mathscr{A}} : \hat{\Omega}_{\mathrm{F}} \times [0,T] \to \Omega_{\mathrm{F}}(t)$$

$$(\hat{x},t) \to x = \hat{\mathscr{A}}(\hat{x},t).$$

We define the fluid domain velocity by

$$\hat{\boldsymbol{w}}(\hat{x},t) = \frac{\partial \hat{\mathscr{A}}}{\partial t}(\hat{x},t).$$

Let $x = \hat{\mathscr{A}}(\hat{x},t)$, we define the ALE derivative:

$$\left.\frac{\partial q}{\partial t}\right|_{\hat{x}}(x,t) = \frac{\mathrm{d}}{\mathrm{d}t} q(\hat{\mathscr{A}}(\hat{x},t),t) = \frac{\partial q}{\partial t}(x,t) + \frac{\partial \mathscr{A}_i}{\partial t}(\hat{x},t)\frac{\partial q}{\partial x_i}(x,t)$$
$$= \frac{\partial q}{\partial t}(x,t) + \boldsymbol{w} \cdot \nabla q(x,t).$$

The displacement of the fluid domain is denoted by $\hat{\boldsymbol{d}}_{\mathrm{F}}$, namely $\hat{\boldsymbol{d}}_{\mathrm{F}}(\hat{x},t) = \hat{\mathscr{A}}(\hat{x},t) - \hat{x}$. The mapping $\hat{\mathscr{A}}$ is typically such that it coincides with $\hat{\boldsymbol{\varphi}}$ on the fluid–structure interface, it satisfies $\boldsymbol{w} \cdot \boldsymbol{n} = 0$ on the artificial boundaries and it is "arbitrary" anywhere else in $\hat{\Omega}_{\mathrm{F}}$.

We denote by $\boldsymbol{\sigma}_{\mathrm{F}}$ the Cauchy stress tensor in the fluid, by $\boldsymbol{\sigma}_{\mathrm{S}}$ the Cauchy stress tensor in the structure, and by $\hat{\boldsymbol{\Pi}} = \hat{J}\hat{\boldsymbol{\sigma}}_{\mathrm{S}}\hat{\boldsymbol{F}}^{-T}$ the first Piola–Kirchhoff tensor.

The load induced by the fluid on the structure is given by

$$\hat{\boldsymbol{f}}_{\Sigma} = \hat{J}\hat{\boldsymbol{\sigma}}_{\mathrm{F}}\hat{\boldsymbol{F}}^{-T}\hat{\boldsymbol{n}}_{\mathrm{S}}. \tag{1}$$

The fluid–structure problem reads:

$$(\mathrm{F}) \begin{cases} \rho_{\mathrm{f}} \left.\frac{\partial \boldsymbol{u}}{\partial t}\right|_{\hat{x}} + \rho_{\mathrm{f}}(\boldsymbol{u} - \boldsymbol{w}) \cdot \nabla \boldsymbol{u} - \operatorname{div} \boldsymbol{\sigma}_{\mathrm{F}} = 0 & \text{in } \Omega_{\mathrm{F}}(t), \\ \boldsymbol{u}(x,t) = \hat{\boldsymbol{u}}_{\mathrm{s}}(\hat{\boldsymbol{\varphi}}_t^{-1}(x),t) & \text{on } \Sigma(t). \end{cases}$$

$$(\mathrm{S}) \begin{cases} \hat{J}\hat{\rho}_{\mathrm{s}} \frac{\partial \hat{\boldsymbol{u}}_{\mathrm{s}}}{\partial t} - \operatorname{div}_{\hat{x}} \hat{\boldsymbol{\Pi}} = 0 & \text{in } \hat{\Omega}_{\mathrm{S}}, \\ \hat{\boldsymbol{\Pi}} \cdot \boldsymbol{n}_{\mathrm{s}} = \hat{J}\hat{\boldsymbol{\sigma}}_{\mathrm{F}}\hat{\boldsymbol{F}}^{-T}\hat{\boldsymbol{n}}_{\mathrm{s}} & \text{on } \hat{\Sigma}. \end{cases}$$

$$(\mathrm{D})\begin{cases}\text{The domain velocity in } \hat{\Omega}_{\mathrm{F}} \text{ satisfies:}\\ \quad \hat{\boldsymbol{w}} = Tr^{-1}(\hat{\boldsymbol{u}}_{\mathrm{s}}|_{\hat{\Sigma}}).\end{cases}$$

where Tr^{-1} denotes an arbitrary extension in the fluid domain. Notice that $\hat{\boldsymbol{d}}_{\Sigma} = \hat{\boldsymbol{d}}_{\mathrm{F}}|_{\Sigma} = \hat{\boldsymbol{d}}|_{\Sigma}$, where $\hat{\boldsymbol{d}}_{\Sigma}$ denotes the displacement of the fluid–structure interface.

This problem can also be written in a variational form:

$$(\mathrm{F})\begin{cases}\text{Find } \boldsymbol{u}(t) \text{ defined on } \Omega_{\mathrm{F}}(t) \text{ such that:}\\ \boldsymbol{u}|_{\Sigma(t)} = \hat{\boldsymbol{u}}_{\mathrm{s}}|_{\widehat{\Sigma}} \quad \text{and} \quad \frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega_{\mathrm{F}}(t)} \rho_{\mathrm{f}}\boldsymbol{u}\cdot\boldsymbol{v}\\ \qquad + \int_{\Omega_{\mathrm{F}}(t)} \mathrm{div}\,[\rho_{\mathrm{f}}\boldsymbol{u}\otimes(\boldsymbol{u}-\boldsymbol{w})]\cdot\boldsymbol{v}\\ \qquad + \int_{\Omega_{\mathrm{F}}(t)} \boldsymbol{\sigma}_{\mathrm{F}} : \nabla\boldsymbol{v} = 0,\\ \quad \text{for all admissible test functions } \boldsymbol{v}.\end{cases}$$

$$(\mathrm{S})\begin{cases}\text{Find } \hat{\boldsymbol{u}}_{\mathrm{s}}(t) \text{ defined on } \widehat{\Omega}_{\mathrm{S}} \text{ such that:}\\ \int_{\widehat{\Omega}_{\mathrm{S}}} \hat{\rho}_{\mathrm{S}}\widehat{J}\frac{\partial\hat{\boldsymbol{u}}_{\mathrm{s}}}{\partial t}\cdot\hat{\boldsymbol{v}} + \int_{\widehat{\Omega}_{\mathrm{S}}} \widehat{\boldsymbol{\Pi}} : \nabla_{\hat{x}}\hat{\boldsymbol{v}} = \int_{\widehat{\Sigma}} \widehat{J}\hat{\boldsymbol{\sigma}}_{\mathrm{F}}\hat{\boldsymbol{F}}^{-T}\hat{\boldsymbol{n}}_{\mathrm{S}}\cdot\hat{\boldsymbol{v}},\\ \quad \text{for all admissible test functions } \hat{\boldsymbol{v}}.\end{cases}$$

$$(\mathrm{D})\begin{cases}\text{Find } \boldsymbol{w}(t) \text{ defined on } \Omega_{\mathrm{F}}(t) \text{ such that:}\\ \quad \boldsymbol{w}|_{\Sigma(t)} = \boldsymbol{u}|_{\Sigma(t)}.\end{cases}$$

Of course, the above equations have to be completed by other boundary conditions depending on the problem at hand.

The fluid is assumed to be incompressible and Newtonian which is generally considered as a reasonable approximation in large arteries. Thus:

$$\boldsymbol{\sigma}_{\mathrm{F}} = -p\boldsymbol{Id} + 2\mu\boldsymbol{\epsilon}(\boldsymbol{u}),$$
$$\mathrm{div}\,\boldsymbol{u} = 0.$$

The structure is assumed to be hyperelastic. Denoting the nonlinear Green–Lagrange strain tensor by $\hat{\boldsymbol{e}} = \frac{1}{2}(\hat{\boldsymbol{F}}^{T}\hat{\boldsymbol{F}} - \boldsymbol{Id})$ and the second Piola–Kirchhoff tensor by:

$$\hat{\boldsymbol{\Sigma}} = \hat{\boldsymbol{F}}^{-1}\hat{\boldsymbol{\Pi}} = J\hat{\boldsymbol{F}}^{-1}\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{F}}^{-T},$$

the constitutive law for the solid reads:

$$\hat{\boldsymbol{\Sigma}} = \frac{\partial\mathscr{W}}{\partial\hat{\boldsymbol{e}}}(\hat{\boldsymbol{e}}), \quad \left(\text{i.e. } \Sigma_{ij} = \frac{\partial\mathscr{W}}{\partial\hat{e}_{ij}}\right),$$

where $\mathscr{W}$ is an elastic energy density (see Section 4.1). We are aware that these constitutive laws are rather rough for arteries which are known to be viscoelastic. We choose them as a first step toward more realistic simulations.

3. Unstructured mesh generation

It will be explained in the next section that it is convenient to use quadrilateral elements for the structure, whereas the fluid domain is generally made of tetrahedra. We address in this section the construction of both meshes.

In the context of numerical simulations, mesh adaptation is often used in order to improve the accuracy of the numerical solution as well as to reduce the number of degrees of freedom (i.e., the mesh size) required to capture the behavior of the solution. Schematically, two approaches can be used to address the problem of constructing an adapted mesh of a given bounded domain:

- mesh optimization using local refinement/derefinement [4,35,9] and
- mesh (re)construction of the domain.

For both cases, numerous algorithms have been proposed and successfully implemented, in particular when isotropic specifications are prescribed.

Nowadays, the generation of unstructured simplicial meshes for domains of arbitrary shape is often considered as a solved problem. Indeed, fast, reliable and fully automatic mesh generation algorithms have been designed over the last two decades and are now routinely used in industrial applications (see [20] for a survey of the main classes of such algorithms). On the other hand, the generation of quadrilateral or hexahedral unstructured meshes has not yet reached the same level of maturity (especially regarding the robustness and the general nature of the algorithms) and therefore, there is still a bit of room for improvement.

Usually, the generation of unstructured quadrilateral meshes receives much attention as it is considered as a pre-requisite for complete hexahedral mesh generation. However, in this study, the coupling between the surface and the volume meshes does not require *strictu sensu* the conformity of both meshes, only the vertices must be matched. Hence, the proposed approach relies on the creation of a quadrilateral surface mesh, each of the quadrilaterals being then subdivided into two triangles that will be preserved during the generation of the tetrahedral volume mesh. The main objective of this section is to briefly present a new indirect approach to produce a quadrilateral mesh for arbitrary curved bounded surfaces.

3.1. Quadrilateral surface meshing

Here, we specifically address the topic of (anisotropic) quadrilateral surface mesh generation using an indirect approach. As such, this approach can be considered as an extension of the method described in [6].

Given a triangular mesh of the domain, the method relies on the principle that a fully quadrilateral mesh can be formed by carefully converting adequate pairs of triangles to form quadrilaterals. By repeating this basic operation over the mesh elements, the whole domain could be covered with quadrilaterals, provided

an even number of triangles. Obviously, such a simple pairing strategy cannot always produce a pure quadrilateral mesh and may lead to a mixed mesh, composed of both triangles and quads. Such a mesh can be further subdivided to result in a quadrilateral mesh.

In order to explain the method employed to convert triangles into quadrilaterals, we need to briefly introduce various notions.

3.1.1. Notion of metric

A metric $\mathscr{M}$, defined by a symmetric positive definite 3×3 matrix associated with the mesh vertices, is introduced to prescribe the desired elements sizes and the aspect ratio of the desired mesh elements. For instance, such a metric can be constructed by an a posteriori error estimate [19] or based on the intrinsic properties of the surface to account for the local curvature [18].

This metric is used to practically compute the edge lengths, the aim being to create a so-called *unit mesh*, i.e. a mesh such that every edge a (considered as a parametrized segment) has a unit length:

$$l(a) = \int_0^1 \sqrt{{}^t\vec{a}\mathscr{M}(t)\vec{a}}\,\mathrm{d}t = 1.$$

The scalar product of two vectors $\vec{a}$ and $\vec{b}$ with respect to the metric $\mathscr{M}$ and the norm of a vector are respectively defined as:

$$\langle \vec{a}, \vec{b}\rangle_{\mathscr{M}} = {}^t\vec{a}\mathscr{M}\,\vec{b}, \quad \|\vec{a}\|_{\mathscr{M}} = [\langle \vec{a}, \vec{a}\rangle]^{\frac{1}{2}}.$$

This allows us to define the angle measure $\widehat{ABC}$ with respect to a metric:

$$\theta_{\mathscr{M}} = \widehat{ABC}_{\mathscr{M}} = \arccos\left(\frac{\langle \overrightarrow{BC}, \overrightarrow{BA}\rangle_{\mathscr{M}}}{\|\overrightarrow{BC}\|_{\mathscr{M}}\|\overrightarrow{BA}\|_{\mathscr{M}}}\right).$$

3.1.2. Quality measure

We can define the quality of a quadrilateral Q = ABCD defined counterclockwise. A quality function ϕ can be related to the measure of the angle and normalized as follows:

$$\phi(\theta_{\mathscr{M}}) = \begin{cases} \frac{2\theta_{\mathscr{M}}}{\pi} & \text{if } 0 \leqslant \theta_{\mathscr{M}} < \frac{\pi}{2}, \\ \frac{2(\pi-\theta_{\mathscr{M}})}{\pi} & \text{if } \frac{\pi}{2} \leqslant \theta_{\mathscr{M}} < \pi, \\ 0 & \text{if } \pi \leqslant \theta_{\mathscr{M}}, \end{cases}$$

and the quality of an element is then defined as the minimum value of the angle quality function ϕ.

This quality function is able to capture all degenerated quadrilaterals (skewed, stretched and concave elements). But obviously, for curved surface, an additional criterion must be added to account for the local curvature. More precisely, coplanar triangles must be paired together preferably so as to preserve the geometric approximation of the surface.

3.1.3. Triangle merging procedure

At first, a unit triangle mesh is created using a classical surface remeshing scheme [18] with respect to the geometric metric based on the local curvatures. Then, starting from the domain boundaries, each pair of triangles is likely to form a new quadrilateral element. Therefore, all such pairs are sorted in decreasing order according to the quality function and are analyzed. Once a pair of triangles has been succesfully checked against the quality criteria, the resulting quadrilateral is formed and the pair is removed from the list. Obviously, isolated triangles may remain at the end of this process. To minimize the number of such triangles, the combination of triangles can be driven by adjacency, from a given quadrilateral.

At completion, a mixed mesh is obtained, containing a few isolated triangles. The latter are then splitted into three quadrilaterals (adding three vertices along the edges and a central vertex on the surface). To preserve the overall mesh conformity, quadrilaterals mesh elements are then splitted accordingly. This operation is not as simple as for planar meshes as additional vertices must be located on the true geometry and not on the straight sided facets.

Notice that this operation leads to a mesh in which all elements have a size two times smaller than expected. To overcome this problem, the initial metric is simply emphasized by a factor of two.

Finally, to improve slightly the overall mesh quality, a node smoothing procedure is carried out over all mesh vertices. It basically consists in moving all vertices in order to optimize the edge lengths with respect to the metric as well as the element shape function quality.

3.2. Tetrahedral volume meshing

Once the quadrilateral surface mesh has been created, each surface quadrilateral is then splitted into two triangles (hence simply changing the mesh topology while preserving the geometry). A tetrahedral mesh is constructed using a Delaunay-based approach [21]. This "classical" procedure is based on three successive stages: vertex insertion, boundary recovery and internal node generation, eventually followed by a mesh quality improvement step. As such, this robust approach preserves the given surface triangulation which, in the context of this study, is a fundamental requirement and prevents the use of other mesh generation methods.

4. Discretization

In our computations, the fluid equations are discretized in space with P1/P1 stabilized finite elements on tetrahedra (or Q1/Q1 on hexahedra). Below, we give some details regarding the structure discretization. We

next present how the load induced by the fluid on the structure is computed and we close this section with the time-advancing schemes.

4.1. Structure: a shell model in large displacements

As explained in the previous section, automatic mesh generation on arbitrary geometries generally produces tetrahedra, and therefore triangles on the surface. Triangular shell elements are thus the most convenient. Our first choice was to use a geometrically exact shell model proposed by Simo et al. [39] with DKT shell elements [8]. This model gave excellent results in other fluid–structure interactions problems (see [25]). In addition, it has been validated on several structural dynamics problems. Nevertheless, the element seems inappropriate for blood flow problems, since in various benchmarks, the results obtained are not in agreement with the expected values. In our attempts to identify the origin of this problem it appears that the actual implementation of the DKT element is very accurate for bending dominated problems but gives poor results for membrane dominated problems. Moreover, its accuracy is extremely sensitive to the shell thickness. This particular behaviour of the shell elements is actually well-known (see chapter 7 in [11]).

To overcome this problem we use the MITC4 general shell element: it is known to be reliable and effective in the two asymptotic states (membrane and bending). The main characteristics of this element are described hereafter. The drawback is the need to use a quadrilateral mesh, that is why a procedure to generate such a mesh from a triangular one has been described in the previous section.

The MITC4 "General Shell Elements" (see [1,11]) can handle large displacements and is interesting as it is based on a general 3D variational formulation. In particular, standard 3D constitutive laws are used. The transversal stress is null and a kinematical constraint makes this model compatible with a Reissner-Mindlin shell model. In our applications, we use a generalized Hook law for which the internal stored energy on the non-deformed configuration is given by:

$$\mathscr{W}(\hat{\boldsymbol{d}}) = \frac{1}{2}\int_{\hat{\Omega}_S} [C^{\alpha\beta\lambda\mu} e_{\alpha\beta}(\hat{\boldsymbol{d}}) e_{\lambda\mu}(\hat{\boldsymbol{d}}) + D^{\alpha\lambda} e_{\alpha z}(\hat{\boldsymbol{d}}) e_{\lambda z}(\hat{\boldsymbol{d}})]\,\mathrm{d}V, \tag{2}$$

where $\boldsymbol{e} = (e_{\alpha\beta})$ denotes the nonlinear Green–Lagrange strain tensor. In Eq. (2) the Greek symbols varying from 1 to 2 are used for the tangential components to the surface, z is the third direction, and

$$C^{\alpha\beta\lambda\mu} = \frac{E}{2(1+\nu)}\left(g^{\alpha\lambda}g^{\beta\mu} + g^{\alpha\mu}g^{\beta\lambda} + \frac{2\nu}{1-\nu}g^{\alpha\beta}g^{\lambda\mu}\right), \tag{3}$$

$$D^{\alpha\lambda} = \frac{8E}{t^2(1+\nu)}g^{\alpha\lambda}, \tag{4}$$

where E is the Young's modulus, ν the Poisson ratio and $g^{\alpha\lambda}$ the contravariant components of the metric tensor.

For the MITC4 finite element the geometry is fully defined by the position of the nodes ($\boldsymbol{x}^{(i)}$) on the mid-surface, the unit outward normal ($\boldsymbol{a}_3^{(i)}$) at the surface in each node and the corresponding thickness ($t^{(i)}$):

$$\boldsymbol{x} = \sum_{i=1}^{N} \lambda_i(r,s)\left(\boldsymbol{x}^{(i)} + z\frac{t^{(i)}}{2}\boldsymbol{a}_3^{(i)}\right), \tag{5}$$

where $\lambda_i(r,s)$ are the standard Q1 Lagrange shape functions in 2D. The displacement is then given by:

$$\hat{\boldsymbol{d}} = \sum_{i=1}^{N} \lambda_i(r,s)\left(\hat{\boldsymbol{d}}^{(i)} + z\frac{t^{(i)}}{2}\boldsymbol{\eta}^{(i)}\right), \tag{6}$$

where $\boldsymbol{\eta}^{(i)}$ corresponds to the variation of the unit vector $\boldsymbol{a}_3^{(i)}$ and verifies the constraint:

$$\boldsymbol{\eta}^{(i)} \cdot \boldsymbol{a}_3^{(i)} = 0.$$

The MITC4 finite element has thus 5 degrees of freedom per node (the three components of the displacement and the two parameters which define the variation of the unit vector). This element is robust and almost free of locking. This desirable feature is obtained by using a particular interpolation strategy for the different components of the strain tensor (see [1]).

4.2. Load computation

To compute the load induced by the fluid on the structure, we use the conservative method proposed in [14]. For the sake of completeness, we briefly recall the main ideas of this approach and show how simple its application is in our case.

To start with, we consider the more general case where the meshes are incompatible, in the sense that the nodes for the fluid and the structure do not match on the interface. We nevertheless assume that the fluid nodes are located on the faces of the structure elements (if this is not the case, one can consider the projection of the fluid nodes onto the surface defined by the structure mesh). The algebraic form of the kinematic condition (F) reads:

$$U_{\mathrm{F},k} = \mathscr{K} U_{\mathrm{S},k} \tag{7}$$

where $U_{\mathrm{F},k}$ (resp. $U_{\mathrm{S},k}$) denotes the vector of the N_Σ^{F} (resp. N_Σ^{S}) degrees of freedom of the kth component of $\boldsymbol{u}$ on the fluid (resp. structure) mesh of Σ, and $\mathscr{K}$ is a $N_\Sigma^{\mathrm{F}} \times N_\Sigma^{\mathrm{S}}$ matrix. The matrix $\mathscr{K}$ can be obtained for example by the mortar element method [5] or by interpolation. In this latter case,

$$\mathscr{K} = \left[\hat{\phi}_j^{\mathrm{S}}(x_i^{\mathrm{F}})\right]_{i=1\ldots N_\Sigma^{\mathrm{F}}, j=1\ldots N_\Sigma^{\mathrm{S}}}$$

where x_i^{F} denotes ith fluid node and $\hat{\phi}_j^{\mathrm{S}}$ the jth basis function of the structure. We now aim at computing

$$F_{S,k} = \left[\int_{\hat{\Sigma}_S} \hat{f}_{\Sigma,k} \hat{\phi}_i^S \right]_{i=1,\dots,N_\Sigma^S},$$

for $k = 1,2,3$, where $\boldsymbol{f}_\Sigma = (f_{\Sigma,k})_{k=1,2,3}$ is defined in (1). Suppose for the moment that the following vector is given:

$$F_{F,k} = \left[\int_{\Sigma_F} f_{\Sigma,k} \phi_j^F \right]_{j=1,\dots,N_\Sigma^F}.$$

Then the discrete energy balance is satisfied as soon as the power of the force computed on the fluid side equals the power computed on the structure side. In other words, if

$$\sum_{k=1}^{3} U_{S,k}^T F_{S,k} = \sum_{k=1}^{3} U_{F,k}^T F_{F,k}.$$

This equality must be true for all vectors satisfying (7). Therefore

$$F_{S,k} = \mathscr{K}^T F_{F,k} \tag{8}$$

We now explain how to compute $F_{F,k}$. Notice that, at the continuous level, if $(\boldsymbol{u},\sigma_F)$ is a solution to (F), if $\boldsymbol{v}$ is a function defined on Σ and $Tr^{-1}(\boldsymbol{v})$ an extension in $\Omega_F(t)$ then

$$\begin{aligned}\int_{\hat{\Sigma}} \hat{\boldsymbol{f}}_\Sigma \cdot \hat{\boldsymbol{v}} = &-\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_F(t)} \rho_f \boldsymbol{u} \cdot Tr^{-1}(\boldsymbol{v}) \\ &- \int_{\Omega_F(t)} \big[\mathrm{div}[\rho_f \boldsymbol{u} \otimes (\boldsymbol{u} - \boldsymbol{w})] \cdot Tr^{-1}(\boldsymbol{v}) \\ &+ \boldsymbol{\sigma}_F : \nabla Tr^{-1}(\boldsymbol{v}) \big].\end{aligned}$$

The vector $F_{S,k}$ can be computed by taking $\boldsymbol{v} = \phi_j^F$ and a discrete counterpart of this relation, consistent with the time scheme (see (9)).

In fact, in view of the special mesh generation techniques adopted in the present work, the nodes of the structure and the fluid do match. Consequently, the matrix $\mathscr{K}$ is simply the identity in our case. The message passing between the two solvers, and more specially the load computation, is therefore straightforward, even if triangles are used on the one side and quadrilaterals on the other one.

4.3. Time discretization

We now describe the time advancing algorithms. We denote by δt the time step and $t^n = n\delta t$. The following quantities are assumed to be known at time t^n: Ω_F^n (approximation to $\Omega_F(t^n)$), $\boldsymbol{u}^n, p^n$, $\hat{\boldsymbol{d}}_F^n, \boldsymbol{w}^n$, $\hat{\boldsymbol{d}}^n, \hat{\boldsymbol{u}}_s^n$. We aim at computing Ω_F^{n+1}, $\boldsymbol{u}^{n+1}$, p^{n+1}, $\hat{\boldsymbol{d}}^{n+1}$ and $\hat{\boldsymbol{u}}_s^{n+1}$.

4.3.1. Fluid domain deformation

We assume for the moment that $\hat{\boldsymbol{d}}_\Sigma^{n+1}$ is known and we compute $\hat{\boldsymbol{d}}_F^{n+1}$ as an arbitrary extension of $\hat{\boldsymbol{d}}_\Sigma^{n+1}$ (for example an harmonic lifting). Then we define the fluid domain velocity by:

$$\hat{\boldsymbol{w}}^{n+1} = \frac{\hat{\boldsymbol{d}}_F^{n+1} - \hat{\boldsymbol{d}}_F^n}{\delta t}$$

and the new domain:

$$\Omega_F^{n+1} = \Omega_F^n + \delta t \boldsymbol{w}^{n+1}$$

We formally denote this step by:

$$\hat{\boldsymbol{d}}_F^{n+1} = \mathscr{D}(\hat{\boldsymbol{d}}_\Sigma^{n+1})$$

(where all the known quantities are omitted for the sake of clarity).

4.3.2. Fluid resolution

We use an implicit Euler scheme for the fluid: we look for $(\boldsymbol{u}^{n+1}, p^{n+1})$ such that, for all admissible test functions $(\boldsymbol{v}, q)$,

$$\begin{cases} \frac{1}{\delta t} \int_{\Omega_F^{n+1}} \rho_f \boldsymbol{u}^{n+1} \cdot \boldsymbol{v} - \frac{1}{\delta t} \int_{\Omega_F^n} \rho_f \boldsymbol{u}^n \cdot \boldsymbol{v} \\ \quad + \int_{\Omega_F^{n+1}} \mathrm{div}(\rho_f \boldsymbol{u}^{n+1} \otimes (\boldsymbol{u}^{n+1} - \boldsymbol{w}^{n+1})) \cdot \boldsymbol{v} \\ \quad - \int_{\Omega_F^{n+1}} p^{n+1} \mathrm{div}\, \boldsymbol{v} + 2 \int_{\Omega_F^{n+1}} \mu \boldsymbol{\epsilon}(\boldsymbol{u}^{n+1}) : \boldsymbol{\epsilon}(\boldsymbol{v}) = 0, \\ \int_{\Omega_F^{n+1}} q \mathrm{div}\, \boldsymbol{u}^{n+1} = 0, \quad \boldsymbol{u}^{n+1}|_{\Sigma^{n+1}} = \frac{\hat{\boldsymbol{d}}_\Sigma^{n+1} - \hat{\boldsymbol{d}}_\Sigma^n}{\delta t}. \end{cases}$$

We formally denote this step by:

$$(\boldsymbol{u}^{n+1}, p^{n+1}) = \mathscr{F}(\hat{\boldsymbol{d}}_F^{n+1})$$

4.3.3. Load computation

According to the discussion of Section 4.2, the force term is straighforwardly obtained by computing the residual of the fluid problem:

$$\begin{aligned}\int_{\hat{\Sigma}} \boldsymbol{f}_\Sigma^{n+1} \cdot \hat{\phi}_i^S = -\Bigg[&\frac{1}{\delta t} \int_{\Omega_F^{n+1}} \rho_f \boldsymbol{u}^{n+1} \cdot \boldsymbol{v}_i - \frac{1}{\delta t} \int_{\Omega_F^n} \rho_f \boldsymbol{u}^n \cdot \boldsymbol{v}_i \\ &+ \int_{\Omega_F^{n+1}} \mathrm{div}(\rho_f \boldsymbol{u}^{n+1} \otimes (\boldsymbol{u}^{n+1} - \boldsymbol{w}^{n+1})) \cdot \boldsymbol{v}_i \\ &- \int_{\Omega_F^{n+1}} p^{n+1} \mathrm{div}\, \boldsymbol{v}_i + 2 \int_{\Omega_F^{n+1}} \mu \boldsymbol{\epsilon}(\boldsymbol{u}^{n+1}) : \boldsymbol{\epsilon}(\boldsymbol{v}_i) \Bigg].\end{aligned} \tag{9}$$

We formally sum up this step by:

$$\boldsymbol{f}_\Sigma^{n+1} = \mathscr{R}(\boldsymbol{u}^{n+1}, p^{n+1})$$

4.3.4. Structure resolution

We use a mid-point scheme for the structure:

$$\begin{cases} \frac{1}{\delta t} \int_{\hat{\Omega}_S} \hat{J} \hat{\rho}_S \hat{\boldsymbol{u}}_s^{n+1} \cdot \hat{\boldsymbol{v}} - \frac{1}{\delta t} \int_{\hat{\Omega}_S} \hat{J} \hat{\rho}_S \hat{\boldsymbol{u}}_s^n \cdot \hat{\boldsymbol{v}} \\ \quad + \frac{1}{2} a_S(\hat{\boldsymbol{d}}^{n+1}, \hat{\boldsymbol{v}}) + \frac{1}{2} a_S(\hat{\boldsymbol{d}}^n, \hat{\boldsymbol{v}}) = \int_{\hat{\Sigma}} \boldsymbol{f}_\Sigma^{n+1} \cdot \hat{\boldsymbol{v}}|_{\hat{\Sigma}}, \\ \frac{\hat{\boldsymbol{d}}^{n+1} - \hat{\boldsymbol{d}}^n}{\delta t} = \frac{\hat{\boldsymbol{u}}_s^{n+1} + \hat{\boldsymbol{u}}_s^n}{2}, \end{cases}$$

with

$$a_S(\hat{\boldsymbol{d}},\hat{\boldsymbol{v}}) = \int_{\hat{\Omega}_S} \hat{\boldsymbol{\Pi}} : \hat{\nabla}_{\hat{x}}\hat{\boldsymbol{v}} = \int_{\hat{\Omega}_S} \frac{\partial \tilde{W}}{\partial \hat{\boldsymbol{F}}}(\hat{\boldsymbol{F}}) : \hat{\nabla}_{\hat{x}}\hat{\boldsymbol{v}}$$
$$= \int_{\hat{\Omega}_S} \frac{\partial \tilde{W}}{\partial \hat{\boldsymbol{F}}}(Id + \nabla_{\hat{x}}\hat{\boldsymbol{d}}) : \hat{\nabla}_{\hat{x}}\hat{\boldsymbol{v}}.$$

We formally sum up this step by:

$$\hat{\boldsymbol{d}}_\Sigma^{n+1} = \mathscr{S}(\boldsymbol{f}_\Sigma^{n+1})$$

4.3.5. Conclusion

The solution to the fluid–structure problem at time step t^{n+1} is achieved by finding a fixed-point $\hat{\boldsymbol{d}}_\Sigma^{n+1}$ of the application $\mathscr{T} = \mathscr{S} \circ \mathscr{R} \circ \mathscr{F} \circ \mathscr{D}$.

5. A Jacobian-free Newton–Krylov algorithm

Classical fixed-point method are very expensive to solve the above problem. A theoretical explanation of this fact is given in [10] in a simplified framework. Thus, we advocate to look for a zero of $\mathscr{A} = \mathscr{I} - \mathscr{T}$ using an inexact Newton algorithm [22,23]:

(1) Initialization: $\hat{\boldsymbol{d}}_0 = \hat{\boldsymbol{d}}^n + \frac{3\delta t}{2}\hat{\boldsymbol{u}}_s^n - \frac{\delta t}{2}\hat{\boldsymbol{u}}_s^{n-1}$.
(ii) Approximate Jacobian resolution: $\tilde{\mathscr{A}}'(\hat{\boldsymbol{d}}_k)\delta\hat{\boldsymbol{d}}_k = -\mathscr{A}(\hat{\boldsymbol{d}}_k)$.
(iii) Update the solution: $\hat{\boldsymbol{d}}_{k+1} = \hat{\boldsymbol{d}}_k + \lambda_k\delta\hat{\boldsymbol{d}}_k$.

In step (ii), $\tilde{\mathscr{A}}'$ denotes a suitable approximation of $\mathscr{A}'$ and in step (iii) λ_k is determined with a linesearch procedure. We choose to solve the linear problem using the Krylov method GMRES [37]. The computation of the Jacobian is therefore not required: it is sufficient to define an approximation of the product of the Jacobian with an arbitrary vector. There exist many techniques to approximate such a product. For example, a finite difference approximation is used in [7]. We refer to the survey paper [24] for a comprehensive presentation of these methods and to [16,27,28] for other attempts at applying Newton algorithms in fluid–structure problems. We follow another route here and we propose to approximate the tangent operator by simplifying the physics. On the one hand, the inexact Newton method may fail to converge if the simplification is too drastic. But on the other hand, if the tangent problem is too rich, the computational time may be as large as with fixed-point methods (even if such inexact Newton algorithms converge in only 2 or 3 iterations).

The main idea of our method is to devise an approximate problem as simple as possible but which takes into account a phenomenon responsible for numerical difficulies: the so-called added-mass effect [10,25]. To define this problem we propose to

- neglect the geometrical variation about the current configuration $\overline{\Omega}_F = \Omega_F(t^n)$,
- linearize the structure about its current state,
- neglect the nonlinear and viscous terms in the fluid.

More precisely, we introduce the simplified fluid problem:

$$\begin{cases} -\triangle p^{n+1} = 0 & \text{on } \overline{\Omega}_F, \\ \frac{\partial p^{n+1}}{\partial n} = -\rho_f \frac{\frac{\hat{\boldsymbol{d}}^{n+1}-\hat{\boldsymbol{d}}^n}{\delta t} - \boldsymbol{u}^n}{\delta t} \cdot \boldsymbol{n} & \text{on } \overline{\Sigma}. \end{cases}$$

As before, this problem defines a formal operator

$$p^{n+1} = \tilde{\mathscr{F}}(\hat{\boldsymbol{d}}_\Sigma^{n+1})$$

Thus, our algorithm reads:

(i) Initialization: $\hat{\boldsymbol{d}}_0 = \hat{\boldsymbol{d}}^n + \frac{3\delta t}{2}\hat{\boldsymbol{u}}_s^n - \frac{\delta t}{2}\hat{\boldsymbol{u}}_s^{n-1}$.
(ii) Resolution of $\tilde{\mathscr{A}}'(\hat{\boldsymbol{d}}_k)\delta\hat{\boldsymbol{d}}_k = -\mathscr{A}(\hat{\boldsymbol{d}}_k)$:
- Evaluate $-\mathscr{A}(\hat{\boldsymbol{d}}_k)$ (with $\mathscr{A} = \mathscr{I} - \mathscr{S} \circ \mathscr{R} \circ \mathscr{F} \circ \mathscr{D}$).
- Solve the linear system with GMRES. The matrix/vector products being evaluated by:

$$\tilde{\mathscr{A}}'(\hat{\boldsymbol{d}}_k)z = z - \mathscr{S}'(\mathscr{F}(\hat{\boldsymbol{d}}_k)) \cdot \tilde{\mathscr{F}}'(\hat{\boldsymbol{d}}_k)z,$$

which only requires one resolution of a scalar Poisson problem and one resolution of the linearized structure (already computed and factorized during the structure solution, since we use a Newton–Raphson with a direct linear solver for the nonlinear elasticity equations).

(iii) $\hat{\boldsymbol{d}}_{k+1} = \hat{\boldsymbol{d}}_k + \lambda_k\delta\hat{\boldsymbol{d}}_k$.

The present method allows a significant reduction of CPU time compared to accelerated fixed-point algorithms [23]. Moreover, according to our experience, it is robust for the physical situation encountered in blood flows: for example, in all our test cases, the linesearch procedure has never been activated (i.e. $\lambda_k = 1$). For the sake of completeness, let us indicate that we use nested preconditioners for the GMRES iterations: the first tangent problem at each time step is not preconditionned (it typically requires 12 GMRES iterations), but the next tangent problems are preconditionned using the Krylov spaces built at the previous inexact Newton steps (consequently, they only require 3 or 4 GMRES iterations). We refer the interested reader to [13] for the details of this technique.

6. Numerical results

The purpose of this section is to illustrate the capabilities of the methods introduced in this article. We basically solve the same physical problem on three different geometries: the fluid is initially at rest and a pressure of $10^3\,\mathrm{N/m^2}$ has been imposed at the inlet for 0.005 seconds. The physical parameters are: $\mu = 0.006\,\mathrm{Ns/m^2}$, $\rho_F = 10^3\,\mathrm{kg/m^3}$, $E = 3\times10^5\,\mathrm{N/m^2}$, $\nu = 0.3$,

$\rho_S = 1.2 10^3 \text{kg/m}^3$ and the thickness of the the shell is 10^{-4}m. The solution is a pressure wave that propagates along the vessel.

The first geometry is an idealized carotid bifurcation for which a hexahedral mesh is available. It will be used as a benchmark. The two other meshes, a cerebral aneurism and another carotid bifurcation, come from medical imaging.

6.1. Comparison hexahedra/tetrahedra

The first mesh of the carotid was generated in hexahedra (by K. Perktold and D. Liepsch). In such a case, the generation of the quadrilateral mesh for the structure is straightforward. We will use these hexahedral/quadrilateral meshes to validate the results obtained after remeshing the volume in tetrahedra and transforming the triangles of the surface in quadrilaterals as described in Section 3 (see Figs. 1 and 2). We have reported on Fig. 3 the norm of the displacement of three arbitrary points located on the common, internal and external carotid (versus time). The curves in solid lines were obtained with the tetrahedral mesh whereas the curves in dots were obtained with the hexahedral one. Notice that the results are in good agreement, which, in some sense, validates our approach.

6.2. Results on geometries coming from medical imaging

In the previous sub-section, we have shown that the results obtained on structured hexahedral/quadrilateral meshes are very close to those obtained on a tetrahedral mesh coupled with a unstructured quadrilateral mesh for the shell. This allows to have a reasonable trust in our approach. In the present section, we show the results of computations performed on geometries coming from medical imaging and for which an hexahedral mesh was not avalaible.

The meshes and the results presented in Figs. 4 and 5 correspond to a congenital cerebral aneurism located at the apex of the bifurcation of a branch of the middle cerebral artery. More details on this aneurism can be found in [38]. The results presented in Figs. 6 and 7 show the propagation of a pressure wave in a carotid bifurcation.

It should be noticed that all the computations have been done by clamping the structure at the inlet and the outlet. These boundary conditions have been chosen for the sake of simplicity but they are clearly not realistic. In particular, if the simulations were performed for larger times, spurious reflexions of the pressure waves would appear. The question of boundary conditions for the fluid–structure problem is a major issue that will not be addressed here. We refer the interested reader to [17] for a discussion on this topic.

Notice that we also observe reflections due to the bifurcation. These reflections, contrarily to the above

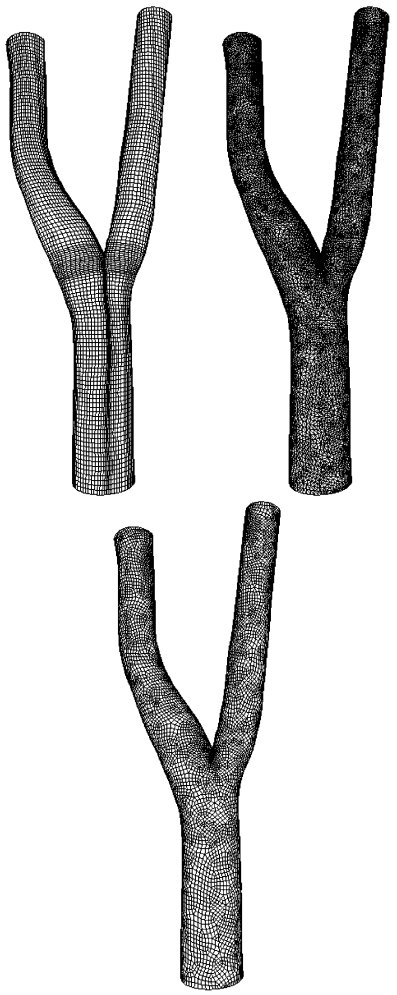

Fig. 1. Hexahedral mesh (by courtesy of K. Perktold and D. Liepsch), tetrahedral mesh, and quadrilateral mesh.

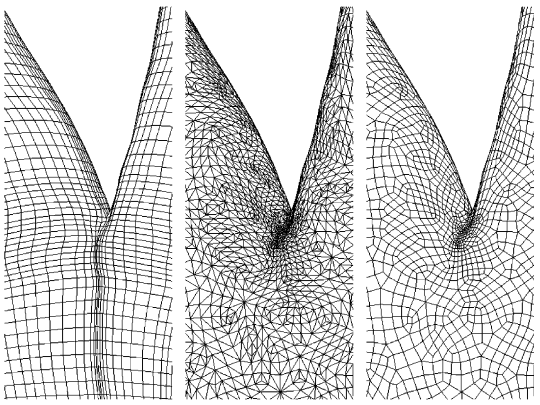

Fig. 2. Magnified view of Fig. 1.

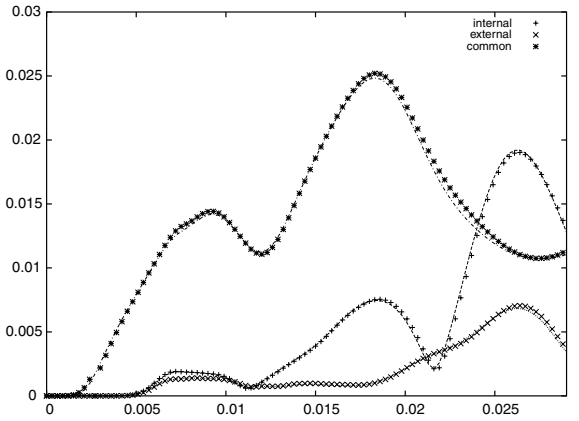

Fig. 3. Norm of the displacement of three arbitrary points versus time. Comparison of the results obtained with structured hexahedral/quadrilateral meshes and unstructured tetrahedral/quadrilaterals meshes (see Figs. 1 and 2).

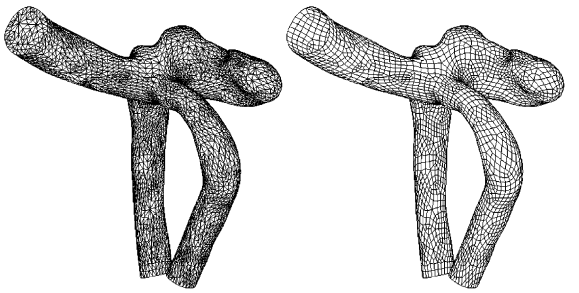

Fig. 4. Fluid and structure meshes of a cerebral aneurism.

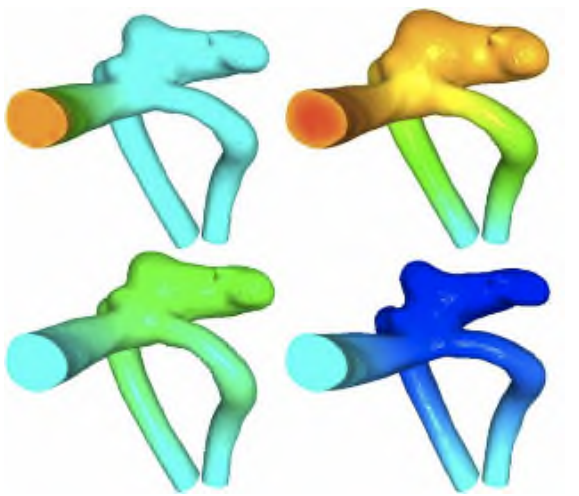

Fig. 5. Propagation of a pressure wave in a cerebral aneurism.

mentioned ones, are meaningful. Simplified 1D fluid–structure models of arteries do not very accurately describe this phenomenon. One possible application,

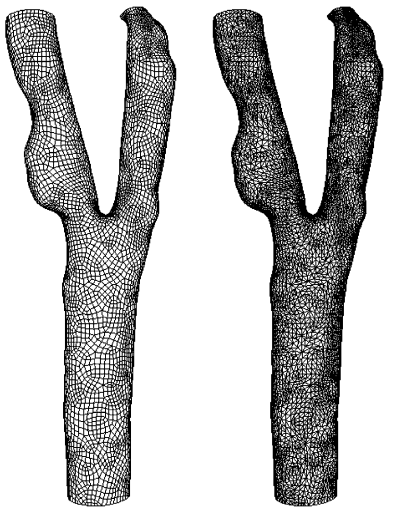

Fig. 6. Fluid and structure meshes of a carotid bifurcation (geometry by courtesy of A. Veneziani and L. Formaggia).

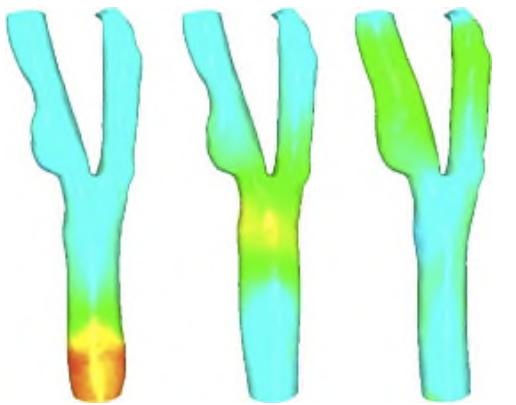

Fig. 7. Propagation of a pressure wave in a carotid bifurcation.

among others, of the present work could be to fit the coefficients of 1D models to correctly account for physiological reflection in a network of arteries.

It is not the purpose of this paper to give biological interpretations of the results. We are aware that there are still other problems to be addressed in order to provide simulations relevant from a medical viewpoint (boundary conditions, constitutive laws, *etc.*). Nevertheless, the algorithms presented here prove to be efficient and reliable to simulate fluid–structure interaction in complex geometries and can be viewed as a first step to further developements.

Acknowledgments

The authors wish to thank M. Thiriet and S. Salmon for the mesh of the aneurism, K. Perkold, D. Liepsch

and A. Leuprecht for the hexahedral mesh of the carotid, L. Formaggia, A. Veneziani and the biomedical group of the CRS4 for the geometry of the second carotid.This work has been partially supported by the *Research Training Network* "Mathematical Modelling of the Cardiovascular System" (HaeMOdel), contract HPRN-CT-2002-00270 of the European Community.

References

[1] Bathe KJ. Finite element procedures. Prentice Hall; 1996.
[2] Bathe KJ, Zhang H. Finite element developments for general fluid flows with structural interactions. Int J Numer Meth Engng 2004, in press.
[3] Bathe KJ, Zhang H, Ji S. Finite element analysis of fluid flows fully coupled with structural interaction. Comput Struct 1999;72(1–3):1–16.
[4] Berger MJ, Jameson A. Automatic adaptive grid refinement for Euler equations. AIAA J 1985;23(4):561–8.
[5] Bernardi C, Maday Y, Patera AT. A new nonconforming approach to domain decomposition: the mortar element method. In: Nonlinear partial differential equations and their applications Collège de France Seminar, Vol XI (Paris, 1989–1991). Pitman Res Notes Math Ser, vol. 299. Longman Sci. Tech.; 1994. p. 13–51.
[6] Borouchaki H, Frey PJ. Adaptive triangular–quadrilateral mesh generation. Int J Numer Meth Engng 1997;41: 915–34.
[7] Brown PN, Saad Y. Convergence theory of nonlinear Newton–Krylov algorithms. SIAM J Optim 1994;4(2): 297–330.
[8] Carrive M, Le Tallec P, Mouro J. Approximation par elments finis d'un modle de coques minces gomtriquement exact. Revue Europenne des Elments Finis 1995;4(5–6): 633–62.
[9] Castro-Diaz MJ, Hecht F, Mohammadi B. New progress in anisotropic grid adaptation for inviscid and viscous flows simulations. In: Proceedings of the 4th International Meshing Roundtable, Albuquerque, USA, October 1995.
[10] Causin P, Gerbeau J-F, Nobile F. Added-mass effect in the design of partitioned algorithms for fluid–structure problems. INRIA Research Report No 5084, 2004.
[11] Chapelle D, Bathe KJ. The finite element analysis of shells—fundamentals. Springer Verlag; 2003.
[12] Deparis S, Fernández MA, Formaggia L. Acceleration of a fixed point algorithm for a fluid–structure interaction using transpiration conditions. Math Model Numer Anal 2003; 37(4):601–16.
[13] Deparis S, Gerbeau J-F, Vasseur X. A dynamic preconditioner for Newton–Krylor Algorithms. Application to Fluid Structure Interaction, INRIA Research Report No 5352, 2004.
[14] Farhat C, Lesoinne M, Le Tallec P. Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: Momentum and energy conservation, optimal discretization and application to aeroelasticity. Comput Meth Appl Mech Engng 1998; 157:95–114.
[15] Farhat C, van der Zee K, Geuzaine Ph. Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear aeroelasticity, in preparation.
[16] Fernández MA, Moubachir M. An exact block-Newton algorithm for the solution of implicit time discretized coupled systems involved in fluid–structure interaction problems. In: Bathe KJ, editor. Second MIT Conference on Computational Fluid and Solid Mechanics. Elsevier; 2003.
[17] Formaggia L, Gerbeau J-F, Nobile F, Quarteroni A. On the coupling of 3D and 1D Navier–Stokes equations for flow problems in compliant vessels. Comp Meth Appl Mech Engng 2001;191(6–7):561–82.
[18] P.J. Frey. About surface remeshing. In Proc.of 9th Int. Meshing Roundtable, New Orleans, USA, 2000.
[19] Frey PJ, Borouchaki H. Surface meshing using a geometric error estimate. Int J Numer Meth Engng 2003;58(2): 227–45.
[20] Frey PJ, George PL. Mesh generation. Application to finite elements. Paris, Oxford: Hermès Science Publ.; 2000.
[21] George PL, Borouchaki H, Frey PJ, Laug P, Saltel E. Mesh generation and mesh adaptivity: theory, techniques. In: Stein E, de Borst R, Hughes TJR, editors. Encyclopedia of computational mechanics. Higher Education Press, John Wiley & Sons; 2004.
[22] Gerbeau J-F. A quasi-Newton method for a fluid–structure problem arising in blood flows. In: Bathe KJ, editor. Proceedings of the second MIT Conference on Computational Fluid and Solid Mechanics. Elsevier; 2003. p. 1355–7.
[23] Gerbeau JF, Vidrascu M. A quasi-Newton algorithm based on a reduced model for fluid–structure interactions problems in blood flows. Math Model Numer Anal 2003; 37(4):631–48.
[24] Knoll DA, Keyes DE. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. J Comp Phys 2004;193:2.
[25] Le Tallec P, Mouro J. Fluid structure interaction with large structural displacements. Comput Meth Appl Mech Engng 2001;190:3039–67.
[26] Lesoinne M, Farhat C. Higher-order subiteration-free staggered algorithm for nonlinear transient aeroelastic problems. AIAA J 1998;36(9):1754–7.
[27] Matthies HG, Steindorf J. How to make weak coupling strong. In: Bathe KJ, editor. Computational Fluid and Solid Mechanics. Elsevier; 2001. p. 1317–9.
[28] Matthies HG, Steindorf J. Partitioned but strongly coupled iteration schemes for nonlinear fluid–structure interaction. Comput Struct 2002;80(27–30):1991–9.
[29] Mok DP, Wall WA. Partitioned analysis schemes for the transient interaction of incompressible flows and nonlinear flexible structures. In: Schweizerhof K, Wall WA, Bletzinger KU, editors. Trends in computational structural mechanics, Barcelona, 2001. CIMNE.
[30] Mok DP, Wall WA, Ramm E. Partitioned analysis approach for the transient, coupled response of viscous fluids and flexible structures. In: Wunderlich W, editor, Proceedings of the European Conference on Computational Mechanics, ECCM'99, TU Munich, 1999.

[31] Mok DP, Wall WA, Ramm E. Accelerated iterative substructuring schemes for instationary fluid–structure interaction. In: Bathe KJ, editor. Computational fluid and solid mechanics. Elsevier; 2001. p. 1325–8.

[32] Morand H, Ohayon R. Interactions fluids–structures. Recherches en Mathématiques Appliquées, vol. 23. Paris: Masson; 1992.

[33] Nobile F. Numerical approximation of fluid–structure interaction problems with application to haemodynamics. PhD thesis, EPFL, Switzerland, 2001.

[34] Piperno S, Farhat C, Larrouturou B. Partitioned procedures for the transient solution of coupled aeroelastic problems. Part I: Model problem, theory and two-dimensional application. Comp Meth Appl Mech Engng 1995; 124:79–112.

[35] Rivara MC. A 3-D refinement algorithm suitable for adaptive and multi-grid techniques. Commn Appl Numer Meth 1992;8:281–90.

[36] Rugonyi S, Bathe KJ. On finite element analysis of fluid flows coupled with structural interaction. CMES—Comp Model Engng Sci 2001;2(2):195–212.

[37] Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Statist Comput 1986;7(3):856–69.

[38] Salmon S, Thiriet M, Gerbeau J-F. Medical image-based computational model of pulsatile flow in saccular aneurisms. Math Model Numer Anal 2003;37(4): 631–48.

[39] Simo JC, Fox DD, Rifai MS. On a stress resultant geometrically exact shell model, Part I : Formulation and optimal parametrization. Comput Meth Appl Mech Engng 1989;72:267–304.

[40] Zhang H, Bathe KJ. Direct and iterative computing of fluid flows fully coupled with structures. In: Bathe KJ, editor. Computational fluid and solid mechanics. Elsevier; 2001. p. 1440–3.