Int. Conf. on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2005
M. Papadrakakis, E. Oñate and B. Schrefler (Eds)
© CIMNE, Barcelona, 2005

MULTI-PHASE FLOW CALCULATION WITH INTERFACE TRACKING COUPLED SOLUTION

Olivier Basset[*], Hugues Digonnet[*], Hervé Guillard[†] and Thierry Coupez[*]

[*] Centre for Material Forming (CEMEF)
Ecole des Mines de Paris, 1 rue Claude Daunesse, 06904 Sophia Antipolis Cedex, France
Email: olivier.basset@ensmp.fr - Web page: http://www-cemef.cma.fr/

[†] MecaGrid Project (INRIA)
INRIA – Sophia Antipolis, BP 93, 06902 Sophia Antipolis Cedex, France
Email: Herve.Guillard@sophia.inria.fr - Web page: http://www-sop.inria.fr/smash/mecagrid/

Key words: Interface Tracking, Multi-phase Flows, Incompressible Navier-Stokes, Pure Advection, Level-Set, Parallel Environment.

Abstract. *This paper is about incompressible Newtonian multi-fluid flows, coupled with an interface tracking method. The presented global approach enables to include different fluids, as well as a transport model, in a unique system. The incompressible flow is computed by using the Navier-Stokes equations solved with a mixed finite element method based on meshes composed of tetrahedra (Mini element with a condensed pyramidal bubble, which recalls the multiscale approaches); and the interface is tracked by using a transport equation. Several stabilization techniques for this transport equation like SUPG, Residual Free Bubbles, and Discontinuous Galerkin are implemented and compared. Moreover, we strongly couple this pure advection equation with the Navier-Stokes system based on a continuous P1 space interpolation. Its uses a combination of Level-Set and a scalar advection equation stabilized with bubbles. Our fully parallel solution enables to treat large scale problems involving moving interfaces that are shown in several examples.*

1 INTRODUCTION

Interface tracking computations are crucial when we talk about multi-phase flow problems: this way, one can observe how the different phases evolve one from another all along the simulation. In the Navier-Stokes equations – used for incompressible Newtonian flows – the interface movement is not explicit; therefore, the need of coupling these equations with some other model appears. Very different numerical methods – like Volume of Fluid, Level-Set or Marker and Cell – have been used for years to keep track of interfaces. In this paper, we focus on a combination of Level-Set and a stabilized scalar advection equation.

This is a complex problem, and it requires big computational resources in order to have accurate results: the model we build does not have to include mesh adaptation [1] (which is almost essential for interface capture with a discontinuous model [2,3,5]) if the domain of calculation is meshed finely enough. That is why our application is parallelized and can be executed on a PC cluster and a computational grid.

The different parts of this paper develop each one of the following aspects: the flow and the transport equations with their finite element formulations; a coupling model of the two problems; the parallel environment used; and finally, some application examples of our coupled problem.

2 INCOMPRESSIBLE NAVIER-STOKES EQUATIONS

2.1 Formulation

The following Navier-Stokes equations are used to compute the incompressible Newtonian flows in a cavity Ω; v represents the velocity field and p the pressure:

$$\begin{cases} \rho \dfrac{dv}{dt} - \nabla \cdot (2\eta\varepsilon(v)) + \nabla p = \rho g \\ \nabla \cdot v = 0 \end{cases} \quad \text{in } \Omega \tag{1}$$

2.2 Variational formulation

The following function spaces are required for the variational formulation of the Navier-Stokes equations:

$$\begin{aligned} V &= \left(H^1(\Omega)\right)^d \\ V^0 &= \left(H^1_0(\Omega)\right)^d \\ P &= L^2(\Omega) \end{aligned} \tag{2}$$

Where $L^2(\Omega) = \left\{ q, \int_\Omega q^2 dV < \infty \right\}$ and $H^1(\Omega) = \left\{ q \in L^2(\Omega), \nabla q \in \left(L^2(\Omega)\right)^d \right\}$

The variational problem consists in finding $(v,p) \in (V,P)$ with $(w,q) \in (V,P)$ such that:

$$\begin{cases} \int_\Omega \rho \dfrac{dv}{dt}.w + \int_\Omega 2\eta\varepsilon(v) : \varepsilon(w) - \int_\Omega p\nabla \cdot w = \int_\Omega \rho g.w \\ \int_\Omega q\nabla \cdot v = 0 \end{cases} \tag{3}$$

2.3 Finite Element formulation

We build finite dimensioned vector spaces V_h and P_h such that $\lim_{h\to 0} V_h = V$ and $\lim_{h\to 0} P_h = P$. The domain Ω is then decomposed in tetrahedra k: $\Omega = \bigcup_K k$ so that we can apply a mixed finite element method using the so called MINI element (P1+/P1) with a continuous interpolation *P1* for the velocity and the pressure, plus a velocity bubble at the center of elements, as shows the figure 1:

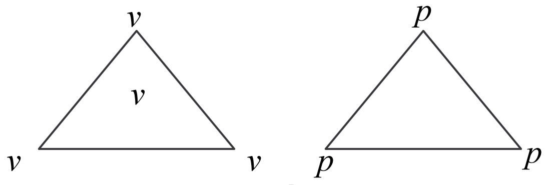

Figure 1: Mini element P1+/P1

The solution (v, p) is then approached by the solution $(v_h, p_h) \in (V_h, P_h)$ of the following problem where $(w_h, q_h) \in (V_h, P_h)$:

$$\begin{cases} \int_\Omega \rho \frac{dv_h}{dt}.w_h + \int_\Omega 2\eta\varepsilon(v_h) : \varepsilon(w_h) - \int_\Omega p_h \nabla \cdot w_h = \int_\Omega \rho g.w_h \\ \int_\Omega q_h \nabla \cdot v_h = 0 \end{cases} \tag{4}$$

2.4 Stabilization

The MINI element is stable in the sense of the Brezzi Babuska conditions. It is close to the stabilized mixed method, the stabilization operator being obtained by a static condensation of a bubble term inside each element. A very important advantage of this approach is the possibility to use an iterative solver for linear system solution. It is based on a MINRES method (Generalized Minimal Residual method) and it shows to be extremely efficient in any case.

Bubbles can play a direct role in stabilizing the Navier-Stokes equations by means of a multiscale[7] approach. Thus, the *resolvable* scales ($\bar{v}_h$) are distinguished from the *unresolvable* scales (b_h) that are, in fact, represented as bubbles. However, the mass conservation requires controlling the bubble L2 norm in contrast to the classical Stokes stabilization, due to the orthogonal properties of the bubble.

$$v_h = \bar{v}_h + b_h \tag{5}$$

When we consider only the Stokes bubble, we have:

$$\begin{cases} \int_\Omega \rho \frac{d\bar{v}_h}{dt}.w_h + \int_\Omega 2\eta\varepsilon(\bar{v}_h) : \varepsilon(w_h) + \int_\Omega 2\eta\varepsilon(b_h) : \varepsilon(w_h) - \int_\Omega p_h \nabla \cdot w_h = \int_\Omega \rho g.w_h \\ \int_\Omega q_h \nabla \cdot \bar{v}_h + \int_\Omega q_h \nabla \cdot b_h = 0 \end{cases} \tag{6}$$

The problem with this last formulation is that it assumes $(\nabla \cdot \bar{v}_h)$ and $(-\nabla \cdot b_h)$ equals.

But, in the case of high Reynolds numbers (big density and small viscosity), $(-\nabla \cdot b_h)$ does not vanish, and therefore, neither does $(\nabla \cdot \bar{v}_h)$. Consequently, the mass conservation equation $(\nabla \cdot \bar{v}_h = 0)$ is not verified, and we need to modify the stabilization as following:

$$\begin{cases} \int_\Omega \rho \frac{d(\bar{v}_h + b_h)}{dt}.w_h + \int_\Omega 2\eta\varepsilon(\bar{v}_h):\varepsilon(w_h) + \int_\Omega 2\eta\varepsilon(b_h):\varepsilon(w_h) - \int_\Omega p_h \nabla \cdot w_h = \int_\Omega \rho g.w_h \\ \int_\Omega q_h \nabla \cdot \bar{v}_h + \int_\Omega q_h \nabla \cdot b_h = 0 \end{cases} \tag{7}$$

This way, we observe that the mass is much better conserved then with the previous stabilization.

3 TRANSPORT EQUATION

3.1 Pure advection equation

For interface tracking purposes, we introduce a phase variable α that demands to be transported along with the computed velocity. As a transport equation, we choose a pure advection scalar equation that is well suited to compute moving liquid when the complex interface shape varies quickly at very large amplitude:

$$\frac{d\alpha}{dt} = \frac{\partial \alpha}{\partial t} + \nabla\alpha.v = 0 \qquad \text{in } \Omega \tag{8}$$

We can use α to approach the characteristic function of the phases, like in the framework of a Volume of Fluid technique. This way, α represents the volume of fluid contained in each element, and thus, is discontinuous between elements. A finite element method of the type of discontinuous Galerkin [2] can be used to solve such a problem with a *P0* interpolation. For years, this technique has been proved to be very effective, especially because of its robustness in terms of time steps and mass conservation. However, it causes a pretty important numerical diffusion, it often demands a mesh adaptation to keep track of an accurate interface [4,5], and its discontinuous interpolation is not very adequate for our coupling model.

Otherwise, we can improve the order of approximation for the interface capture by using a continuous *P1* space interpolation. It is then interesting to have a regular α, like a Level-Set function, that is initialized as a distance function to the interface: $\alpha(x) = dist(x, \Gamma)$. Thus, the zero isosurface $\alpha(x) = 0$ represents perfectly the interface. After having initialized the phase variable, we can transport α and then conserve the interface $\alpha(x) = 0$; but the Level-Set function itself is not conserved. At this point, we can either use a full Level-Set method (and reinitialize the function a every time increments) or apply an upwind effect to the advection by using a method like SUPG. In this paper, we focus on this last point: an association of a Level-Set – like function and a SUPG – like stabilization method.

Furthermore, with a continuous interpolation *P1*, we solve for one scalar per node in the mesh, whereas, with a discontinuous interpolation *P0*, we solve for one scalar per element that are much more numerous than the nodes (about 2 times more in 2D and 5 or 6 times more in 3D). That makes the resolution much faster: a continuous resolution takes 14 seconds while a discontinuous resolution takes 63 seconds. A last, but not least, the memory needed in the linear system is much larger in the discontinuous case. Then, we are more interested in a continuous interpolation *P1*.

3.2 Variational and finite element formulations

The variational problem consists in finding $\alpha \in H^1(\Omega)$ with $\phi \in H_0^1(\Omega)$ such that:

$$\int_\Omega \frac{\partial \alpha}{\partial t}.\phi + \int_\Omega \nabla \alpha . v . \phi = 0 \tag{9}$$

With the same discretization as for the Navier-Stokes equations, α is approached by the solution α_h of this problem:

$$\left(\frac{\partial \alpha_h}{\partial t}, \phi_h\right)_\Omega + \left(\nabla \alpha_h . v, \phi_h\right)_\Omega = 0 \tag{10}$$

3.3 Stabilization

While finite element methods (and the classical Galerkin formulation naturally associated with them) are well suited for solving incompressible fluid flows, they are somehow inappropriate for purely convective problems. The reason is that the central differencing property of the standard Galerkin method causes spurious oscillations in the solution when the problem is not dominated by diffusion, but by advection.

Firstly, we observe that a standard Galerkin method gives much better results (and less oscillations in the solution) when the function α to be transported is initialized as a Level-Set function; i.e. a distance function to the fluid interface.

Secondly, we can apply stabilization by keeping the bubble and multiscale philosophy [8], as for the Navier-Stokes equations. This way, the method like the Residual-Free Bubbles [6,8,9] provides us the wanted stabilization for the advection problem:

$$\left(\frac{\partial \alpha_h}{\partial t}, \phi_h\right)_\Omega + \left(\nabla \alpha_h . v, \phi_h\right)_\Omega + \left(\left(\frac{\partial \alpha_h}{\partial t} + \nabla \alpha_h . v\right) \tau_a \left(\frac{\partial \phi_h}{\partial t} . + \nabla \phi_h . v\right)\right)_\Omega = 0 \tag{11}$$

$$\tau_a = \frac{1}{3}\frac{h_k^v}{|v_k|}$$

$|v^k|$ is the average velocity norm in the element *k*, and h_k^v is the length of the longest segment parallel to v^k and contained in *k*.

Note: in practice, we find that a semi-implicit model works better, especially in terms of mass conservation. Thus, we define the term $(\nabla \alpha_h .v)$ to be $1/2(\nabla \alpha_h .v)+1/2(\nabla \alpha^- .v)$, where α^- represents the known solution of the previous time increment.

4 MULTI-PHASE MODELLING

4.1 Interface capture

This paragraph shows a way to couple the Navier-Stokes equations with the scalar advection equation in order to simulate multi-phase flows. The issue here is to capture effectively the interfaces between phases.

Since, as seen previously, it is better to transport a Level-Set – like function with our continuous (*P1* interpolation at the nodes) formulation, we initialize α as a signed distance function to the interface:

$$\begin{cases} \alpha > 0 & \text{in the first fluid} \\ \alpha < 0 & \text{in the second fluid} \\ \alpha = 0 & \text{at the interface} \end{cases} \tag{12}$$

As a matter of fact, this initialization technique provides an almost perfect initial interface definition: the isosurface of α at zero shows perfect planes and curves (see figure 2). This way, we gain precision versus discontinuous methods that define phases inside elements [4,5].

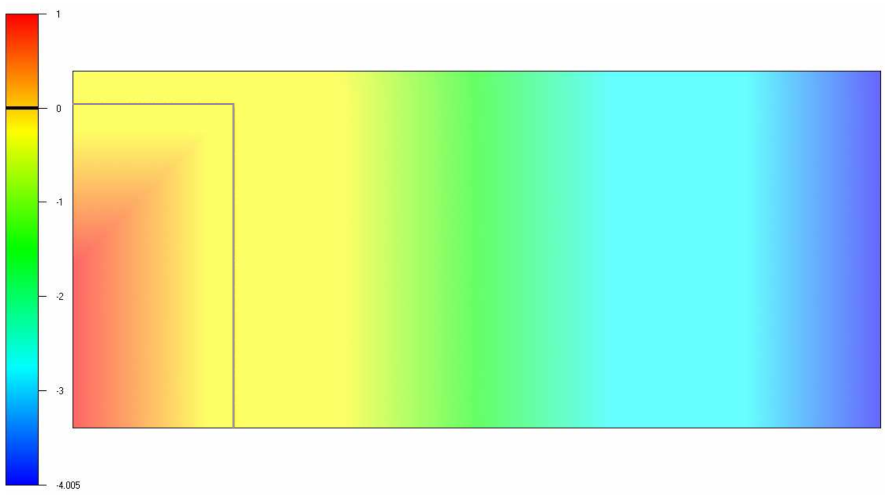

Figure 2: Level-Set initialization of α in 2 dimensions

Then, when this function is transported during the simulation, the isosurface zero of α is kept to be the fluid interface. Moreover, thanks to the fact that α is signed, the different fluids present in the flow can be located by using an appropriate mix law presented in the next section. For sake of simplicity, this paper only considered bi-fluid flows, but the presented methods can easily be extended to a higher number of different fluids present in the flow.

4.2 A mix law for multi-phase modeling

Since we have several phases and only one flow solver, we need to turn the different viscosities and densities into homogeneous parameters to use during resolution of the Navier-Stokes system. Thus, based on a mix law, we make η and ρ depend on the characteristics of each fluid: for a simulation involving a first fluid with viscosity η_1 and density ρ_1, and a second fluid with viscosity η_2 and density ρ_2, we define the parameters η and ρ in the Navier-Stokes equations as following:

$$\begin{gathered}\eta = \eta_1 f(\alpha) + \eta_2 (1 - f(\alpha)) \\ \rho = \rho_1 f(\alpha) + \rho_2 (1 - f(\alpha)) \\ f(\alpha) \in [0,1]\end{gathered} \tag{13}$$

With such a definition, we have $\eta = \eta_1$ and $\rho = \rho_1$ in the first fluid; $\eta = \eta_2$ and $\rho = \rho_2$ in the second fluid; and intermediate values inside a given mix zone. We will see that this buffer zone where the two fluids coexist at the same time can be as small as the element size. This way, the diffusivity goes no larger then the mesh size all along the simulation.

$f(\alpha)$ can be straight or not, abrupt or gradual, etc...; and that determines the nature and the size of the buffer zone. A judicious choice is to set this buffer zone only inside some wanted elements. In fact, such a function decides the amount of all fluids present inside elements crossed by the interface; while other elements are all fully filled by one fluid or another. A big benefit would be that the zone where several fluids coexist is totally controlled, and cannot grow larger then the element size. By looking at the numerical diffusion brought naturally by pure advection equations, this last point is significant.

The section 6 shows applications involving two phases with very different parameters (local Reynolds can reach about 10 000), and we can see that the propose mix law manage efficiently this difference.

4.3 Coupled formulation

The two models we seek to couple are both using continuous discretizations (*P1* interpolation), so we can solve (v, p, α) simultaneously in a single system, without having to include another model like mesh adaptation. Consequently, the resulting strong coupled system contains five unknowns per node of the meshed cavity (in three dimensions). Thus, the final coupled formulation looks like:

$$\begin{cases} \rho \dfrac{dv}{dt} - \nabla \cdot (2\eta\varepsilon(v)) + \nabla p = \rho g \\ \nabla \cdot v = 0 \\ \dfrac{d\alpha}{dt} = 0 \\ \eta = \eta_1 f(\alpha) + \eta_2 (1 - f(\alpha)) \\ \rho = \rho_1 f(\alpha) + \rho_2 (1 - f(\alpha)) \end{cases} \tag{14}$$

The system (14) being defined globally in the whole domain Ω, the test functions w, q and ϕ of the corresponding weak formulation are required to vanish at the boundary of the domain, but not at the interface of the fluids:

$$\begin{cases} \int_\Omega \rho \dfrac{d(\bar{v}_h + b_h)}{dt}.w_h + \int_\Omega 2\eta\varepsilon(\bar{v}_h) : \varepsilon(w_h) + \int_\Omega 2\eta\varepsilon(b_h) : \varepsilon(w_h) - \int_\Omega p_h \nabla \cdot w_h = \int_\Omega \rho g.w_h \\ \int_\Omega q_h \nabla \cdot \bar{v}_h + \int_\Omega q_h \nabla \cdot b_h = 0 \\ \int_\Omega \dfrac{\partial \alpha_h}{\partial t}.\phi_h + \int_\Omega \nabla \alpha_h.v.\phi_h + \int_\Omega \left(\dfrac{\partial \alpha_h}{\partial t} + \nabla \alpha_h.v \right) \tau_a . \left(\dfrac{\partial \phi_h}{\partial t}. + \nabla \phi_h.v \right) = 0 \\ \eta = \eta_1 f(\alpha) + \eta_2 (1 - f(\alpha)) \\ \rho = \rho_1 f(\alpha) + \rho_2 (1 - f(\alpha)) \end{cases} \tag{15}$$

5 PARALLEL ENVIRONMENT

The finite element solver described above has been implemented in parallel by using the Message Passing Interface (MPI). The local matrices and second members are injected in the PETSc libraries, which solve the linear system of five unknowns per node (in three dimensions) with a Generalized Minimal Residual method.

First, we executed our parallel code on a PC cluster, and then, on a computational grid made of several linked clusters. Thus, we have the opportunity to run some large application cases that will be presented in the last section of this paper.

5.1 PC cluster

The PC cluster we use here contains 32 Pentium IV (2.8 GHz) bi-processors with 2 GB of RAM and a Myrinet network at 2 Gb/s. Our application has a very good parallel efficiency: a simulation that takes 1 day, 18 hours and 36 minutes on 1 processor, takes 1 hour and 19 minutes on 32 processors. That makes an acceleration of 32.4 on 32 processors (note that the cluster has a non-exclusive usage, which can introduce some random fluctuations in timings).

Next, we were able to execute a case of multi-phase flow in a 3 dimensioned cavity meshed pretty finely with 2 148 355 nodes and 12 418 472 elements. During this simulation, 600 time increments were necessary, and 6 billions and 450 millions unknowns were solved at each increment. The computations lasted for 5 days and 1 hour on 32 processors of our PC cluster. Note that, inside a single cluster, performances are supposed homogeneous, and then, a standard uniform partitioning is enough to make a good load balancing among the processors: in this case the amount of work is simply divided by the number of processors.

5.2 Computational grid

In the framework of the MecaGrid project, a computational grid has been built by connecting the PC clusters of several sites. This grid is dedicated to computations in heterogeneous fluid mechanics. The performances in terms of computational power and network speed are considerably heterogeneous throughout the grid: not only are the clusters themselves different, but also the internet links between them are very different and somehow less effective then the network connections inside each cluster. Therefore, we need to generate a partition that takes into account this kind of information, so that processors and networks that are less efficient do not slow down others too much. Our partitioning optimization tool divides the mesh in an effective way such that powerful processors have more work to do than others. In addition, the interfaces between partitioned mesh parts are minimized in function of the network speed. This way, the better a network between two processes is, the more communications there are, compared with other slower pairs of processors.

Furthermore, other techniques can improve even more the grid usage: since we are penalized by bad communication between sites due to a basic internet connection (with some random fluctuations in speed), we prefer to give more computation work to the processors in order to lower the amount of communications. Thus, with the PETSc library, we apply different degrees k to the ILU(k) preconditioner, before the linear system is solved by an iterative method. For instance, compared with the ILU(0), ILU(1) allows more nonzero entries in the matrices, but needs less iterations to converge towards the solution of the linear system. As a result, processors have more computations to make, but communicate less with others using MPI messages. However, the memory needed by processes is significantly larger. With this technique, we are able to improve the simulation time of about 40%, which is not negligible.

6 VALIDATIONS AND APPLICATIONS

In this section, we validate the choices made during the formulation part of this paper (paragraphs 2 and 3), especially by looking at the spurious oscillations that appear in the solution of the transport equation, and the mass conservation offered by both pure advection and Navier-Stokes equations. After, the falling fluid column test case is presented in two and three dimensions.

6.1 Pure advection: Oscillations

The test case we use here is a fluid column falling under its own weight, with a small 2D mesh of 2 205 nodes and 4 408 elements. The association of the Navier-Stokes equations and a transport equation solved by different methods, allows us to make some comparisons in terms of oscillations caused by the pure advection equation. The table 1 shows the maximum oscillations appearing in the solution after 100 time increments caused by a discontinuous Galerkin, a continuous standard Galerkin, a Level-Set, a SUPG and a Residual Free Bubble (RFB) method (discussed in 3.2):

Method	Discontinuous	Standard Galerkin	LevelSet	SUPG	RFB
Oscillations %	0	180	0.4	1	0.6

Table 1: Transport equation: Oscillations in the solution

A 0% oscillation means that a scalar $\alpha \in [-1,1]$ that is transported stays between -1 and 1; and a 100% oscillation means that a scalar $\alpha \in [-1,1]$ that is transported goes from -2 to 2 at the end of the simulation.

Although a discontinuous method causes no oscillation, a Standard Galerkin provokes a lot of spurious oscillations. But, stabilization techniques like Level-Set, SUPG or RFB reduce significantly the unwanted noise in the solution.

6.2 Pure advection: Mass conservation

At the beginning of the simulation, the fluid column has a total mass of 2; and the table 2 shows the remaining mass of fluid after 100 time increments. A perfectly conservative method must show a mass of 2 all along the simulation. Here are compared several methods to solve the scalar advection equation: the discontinuous Galerkin and the standard Galerkin methods; and a standard Galerkin method with a semi-implicit time scheme (discussed in 3.3):

Method	Discontinuous	Standard Galerkin	Semi-mplicit
Mass remaining	1.986	1.853	1.964
Pourcentage lost	1.4	14.7	3.6

Table 2: Transport equation: Mass of fluid remaining after 100 time increments

We can see that different finite element methods show various results: while a discontinuous Galerkin method is well conservative, the standard Galerkin does seem to conserve the mass effectively. However, using a semi-implicit time scheme helps a lot, and conserves the mass in a much better way.

6.3 Navier-Stokes equations: Mass conservation

Our discontinuous method to solve the pure advection scalar equation is known to conserve efficiently the mass and to have a good solution without any spurious oscillation.

Then, a decoupled model of the Navier-Stokes equations and this discontinuous transport can be used for comparison purposes about the Navier-Stokes stabilization. The table 3 contains the amount of fluid that remains after 100 time increments by using the stabilizations of equations 6 and 7:

Stabilization	Equation 6	Equation 7
Mass remaining	1.767	1.986
Pourcentage lost	23.3	1.4

Table 3: Navier-Stokes stabilization: Mass of fluid remaining after 100 time increments

We observe that the Navier-Stokes equations with a basic Stokes stabilization (equation 6) loses 23.3% of the mass after 100 increments, while our Navier-Stokes stabilization (equation 7) is much more conservative and loses only 1.4% of mass after 100 time increments.

6.4 The falling fluid column in 2D

This test case is often used to validate study of non stationary fluid flows with a moving free surface. A fluid column gets crushed under its own weight and surges from wall to wall. The air that surrounds the fluid in the cavity is considered to be an incompressible fluid with smaller viscosity and density. Thus, the behavior of the fluid approaches a free surface flow where local Reynolds can reach about 10 000.

The figure 3 shows the presence of the fluid at six given moments during the simulation. The function α was initialized as in the figure 2 (see paragraph 4.1), and transported with the flow velocity. The fluid (here in a red color) has a higher density and viscosity then the second fluid (in blue on the picture). The mixed zone (in green), where the two fluids coexist at the same time, is not larger than the elements crossed by the interface.

In this example, the mesh contains 35 364 nodes and 70 762 elements, and 2 000 time increments were needed for a 5 seconds simulation time. It was executed on 16 processors of a PC cluster and lasted for about 2 hours.

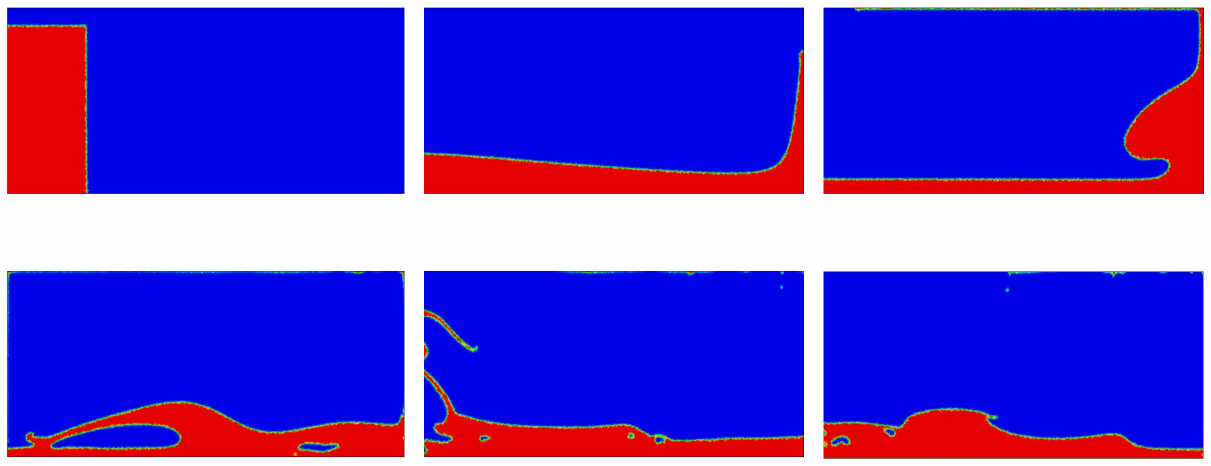
Figure 3: Falling fluid column in 2D

6.5 The falling fluid column in 3D

This example is the same as the previous one, but in 3 dimensions. The 3D cavity is meshed with 2 148 355 nodes and 12 418 472 elements, and 600 time increments were needed for a 3 second simulation time. The computation lasted for 5 days on 32 processors.

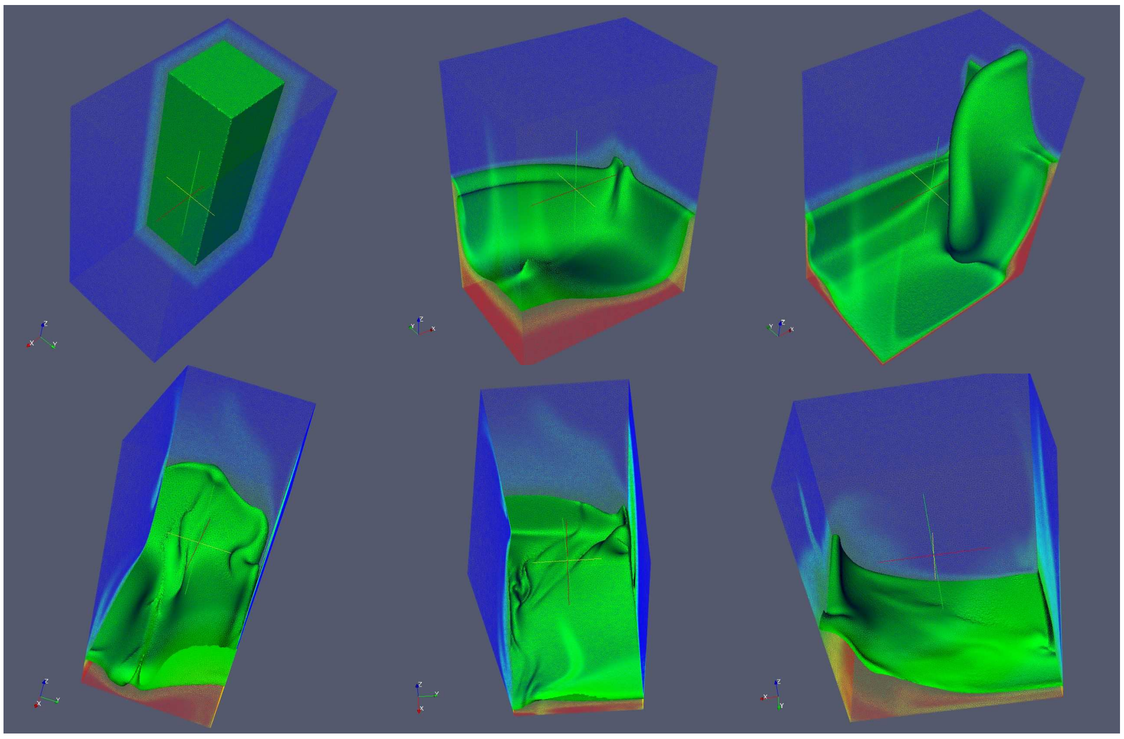

Figure 4: Falling fluid column in 3D

7 CONCLUSIONS

We describe a general approach based on a stabilized finite element method that allows us to treat large scale problems involving moving interfaces. The stabilization has been performed mainly throughout the multiscale and residual free bubble methods for both the incompressible Navier-Stokes equations and the pure advection equation.

An interface tracking of a fluid column falling under its own weight is proposed with a mesh of over 2 million nodes (and 12 million elements).

REFERENCES

[1] T. Coupez, “A mesh improvement method for 3D automatic remeshing”, in N. P. Weatherill et al., *Numerical Grid Generation in Computational Fluid Dynamics and Related Fields*, Pineridge Press, 615-626, (1994).

[2] T. Coupez, and E. Pichelin, “Solution of the 3d mold filling by a Taylor Discontinuous Galerkin Method”, in proc. of *World Congress on Computational Mechanics*, CDRom CIMNE, Buenos Aires, (1998).
[3] E. Bigot, and T. Coupez, “Capture of 3D moving free surfaces and material interfaces by mesh deformation”, in proc. of *ECCOMAS 2000*, CDRom, Barcelona (2000).
[4] T. Coupez, J. Bruchon, and S. Batkam, “Complex free surface and interface dynamics in material forming computation”, in proc. of fifth *World Congress on Computational Mechanics*, Vienna, Austria, (2002).
[5] T. Coupez, C. Gruau, Ch. Pequet, and J. Bruchon, “Metric map and anisotropic mesh adaptation for static and moving surfaces”, in proc. of sixth *World Congress on Computational Mechanics*, Beijing, China, (2004).
[6] F. Brezzi and A. Russo, “Choosing bubbles for advection-diffusion problems”, *Math. Meth. Models Meth. App. Sci.*, **4**, 571-587, (1994).
[7] T.J.R Hughes, “Multiscale phenomena: Green’s functions, the Dirichlet to Neuman formulation, subgrid scale models, bubbles and the origin of stabilized methods”, *Comput. Meth. Appl. Mech. Engrg.*, **127**, 387-401, (1995).
[8] F. Brezzi, L. P. Franca, T. J. R. Hughes, and A. Russo, “$b = \int g$”,*Comput. Meth. Appl. Mech. Engrg.*, **145**, 329-339, (1997).
[9] F. Brezzi, L.P. Franca, and A. Russo, “Further considerations on residual-free bubbles for advective-diffusive equations”, *Comput. Meth. Appl. Mech. Engrg.*, **166**, 25-33, (1998).