3D simulation of two-fluid flows using a finite element/level-set method in 3D. Application to two drop benchmarks

Thibaut Métivet, Vincent Chabannes, Vincent Doyeux, Mourad Ismail, Christophe Prud'Homme

3D simulation of two-fluid flows using a finite element/level-set method. Application to two drop benchmarks.

T. Metivet[a,b,*], V. Chabannes[c], V. Doyeux[d], M. Ismail[a], C. Prud'homme[c]

[a] *Université Grenoble-Alpes / CNRS, Laboratoire Interdisciplinaire de Physique / UMR 5588 Grenoble, F-38041, France*
[b] *Université Grenoble-Alpes / CNRS, Laboratoire Jean Kuntzman / UMR 5224. Grenoble, F-38041, France*
[c] *Université de Strasbourg / CNRS, IRMA / UMR 7501, Cemosis - Center of Modeling and Simulation, Strasbourg, F-67000, France*
[d] *Institut de Mécanique des Fluides de Toulouse / CNRS-INPT-UPS UMR 5502, Université de Toulouse, F-31400 Toulouse, France*

Abstract

We present a numerical framework for the simulation of three-dimensional multi-fluid flows based on a finite element/level-set approach. The method allows a full Eulerian "tracking" of the interfaces between the fluids, and the properties of the interfaces can be directly taken into account as surface forces. The resolution of the fluid equations and the advection of the interface can be easily decoupled, which enables the use of efficient solving strategies. We also present a 3D benchmark of the rise of a drop in a viscous fluid. We use two different setups and compare our results to previous results obtained with other approaches to validate our method.

Keywords: Navier-Stokes, finite element method, level-set method, fast-marching, Hamilton-Jacobi redistanciation method, two-fluid flows, high-order level-set

Introduction

1. Level set description

1.1. Description

Let $\Omega \subset \mathbb{R}^p$ $(p = 2, 3)$ be a bounded domain decomposed into two distinct fluid subdomains Ω_1 and Ω_2. Denote Γ the interface between the two subdomains. In order to track the interface $\Gamma(t)$, which is moving at some velocity $\boldsymbol{u}$, we use the level set method, which provides a way to implicitly follow the interface position over time while naturally handling possible topological changes. The level set method, described in [1–3], features a continuous scalar function ϕ (the *level set* function) defined on the whole domain. This function is chosen to be positive in Ω_1, negative in Ω_2 and zero on Γ. The motion of the interface is described by the advection of the level set function by the divergence-free velocity field $\boldsymbol{u}$:

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla \phi = 0, \quad \nabla \cdot \boldsymbol{u} = 0. \tag{1}$$

The signed distance to the interface function turns out to be a convenient choice for ϕ, as the intrinsic property $|\nabla \phi| = 1$ eases the numerical resolution of the advection equation and the regularity of the distance function allows us to use ϕ as a support for delta and Heaviside functions (c.f. section 1.2).

Nevertheless, it is known that the advection equation (1) does not preserve the property $|\nabla \phi| = 1$ and it is necessary to reset $\phi(t)$ to a distance function without moving the interface, [4–6]. To reinitialize $\phi(t)$ and enforce $|\nabla \phi(t)| = 1$, we can either solve a Hamilton-Jacobi equation, which "transports" the

*Corresponding author
Email address: thibaut.metivet@univ-grenobles-alpes.fr (T. Metivet)

isolines of ϕ to their proper positions, or use the fast marching method, which resets the values of ϕ to the distance to the interface from one degree of freedom to the next, starting from the interface (see [7] for further details).

1.2. Interface related quantities

In two-fluid flow simulations, we need to define interface-related quantities such as the density, the viscosity, or interfacial forces. To this end, we introduce the smoothed Heaviside and delta functions :

$$H_\varepsilon(\phi) = \begin{cases} 0, & \phi \le -\varepsilon, \\ \frac{1}{2}\left[1 + \frac{\phi}{\varepsilon} + \frac{\sin(\frac{\pi\phi}{\varepsilon})}{\pi}\right], & -\varepsilon \le \phi \le \varepsilon, \\ 1, & \phi \ge \varepsilon. \end{cases} \qquad \delta_\varepsilon(\phi) = \begin{cases} 0, & \phi \le -\varepsilon, \\ \frac{1}{2\varepsilon}\left[1 + \cos\left(\frac{\pi\phi}{\varepsilon}\right)\right], & -\varepsilon \le \phi \le \varepsilon, \\ 0, & \phi \ge \varepsilon. \end{cases}$$

with ϕ the distance to the interface function, and ε a parameter controlling the "numerical thickness" of the interface. Typically, we choose $\varepsilon \sim 1.5\ h$, with h the mesh size of the elements crossed by the 0 iso-value of the level set function.

The Heaviside function is used to define physical quantities which have different values on each subdomain. For example, we define the density of two-fluid flow as $\rho = \rho_2 + (\rho_1 - \rho_2)H_\varepsilon(\phi)$ (we use a similar expression for the viscosity μ). The delta function allows to define quantities on the interface, in particular in the variational formulations, where we replace integrals over the interface Γ with integrals over the entire domain Ω using the smoothed delta function: if ϕ is a signed distance function (i.e. $|\nabla\phi| = 1$), we have $\int_\Gamma 1 \simeq \int_\Omega \delta_\varepsilon(\phi)$.

However, as mentioned above, the advection equation does not preserve the property $|\nabla\phi| = 1$, and the level set function ϕ is therefore not exactly a distance function. The support of δ_ε can then have a different size on each side of the interface, as it is narrowed in the regions where $|\nabla\phi| > 1$ and enlarged in those where $|\nabla\phi| < 1$.

Fortunately, as suggested in [8], $\frac{\phi}{|\nabla\phi|}$ is kept close to a distance function near the interface, and it has the same 0 iso-value as ϕ. We can thus use $\frac{\phi}{|\nabla\phi|}$ as a support for interface-related functions to ensure one recovers the correct limit as $\varepsilon \to 0$. The interfacial integral then reads

$$\int_\Gamma f \simeq \int_\Omega f\ \delta_\varepsilon\left(\frac{\phi}{|\nabla\phi|}\right)$$

and the density and viscosity in the domain are defined as

$$\begin{aligned} \rho_\phi &= \rho_2 + (\rho_1 - \rho_2)\ H_\varepsilon\left(\frac{\phi}{|\nabla\phi|}\right) \\ \mu_\phi &= \mu_2 + (\mu_1 - \mu_2)\ H_\varepsilon\left(\frac{\phi}{|\nabla\phi|}\right). \end{aligned} \tag{2}$$

2. Numerical implementation

2.1. Discretization framework

The numerical implementation is performed using the Feel++ — finite element C++ library — [9–11]. Feel++ allows to use a very wide range of Galerkin methods and advanced numerical methods such as domain decomposition methods including mortar and three fields methods, fictitious domain methods or certified reduced basis. The ingredients include a very expressive embedded language, seamless interpolation, mesh adaption and seamless parallelization. It has been used in various contexts including the development and/or numerical verification of (new) mathematical methods or the development of large multi-physics applications [12–14]. The range of users span from mechanical engineers in industry,

physicists in complex fluids, computer scientists in biomedical applications to applied mathematicians thanks to the shared common mathematical embedded language hiding linear algebra and computer science complexities.

Feel++ provides a mathematical kernel for solving partial differential equation using arbitrary order Galerkin methods (FEM, SEM, CG, DG, HDG, CRB) in 1D, 2D, 3D and manifolds using simplices and hypercubes meshes [9–11] : *(i)* a polynomial library allowing for a wide range polynomial expansions including H_{div} and H_{curl} elements, *(ii)* a lightweight interface to BOOST.UBLAS, EIGEN3 and PETSC/SLEPC as well as a scalable in-house solution strategy *(iii)* a language for Galerkin methods starting with fundamental concepts such as function spaces, (bi)linear forms, operators, functionals and integrals, *(iv)* a framework that allows user codes to scale seamlessly from single core computation to thousands of cores and enables hybrid computing.

We work within the continuous Galerkin variational formulation framework, and use Lagrange finite elements to spatially discretize and solve the equations governing the evolutions of the fluid and the level set. Temporal discretization is performed using a Backward Differentiation Formula of order two for the time derivatives when applicable, falling back to an order one Euler formula when the two previous steps are not available. We discuss later in section 3.2 the solution strategy used within our framework.

2.2. Levelset advection

Due to its hyperbolic nature, the level set advection equation is subjected to spurious oscillatory instabilities when solved within a standard finite element framework. To circumvent this well-known problem and stabilize the resolution of the discrete advection equation, we use the Galerkin Least Square (GLS) approach, introduced in [15] and studied in [16] for the case of advection-diffusion equations.

It consists in introducing a stabilization term in the discrete Galerkin variational formulation, which then reads:

Find $\phi_h \in \mathcal{P}_h^k$ such that $\forall\, \psi_h \in \mathcal{P}_h^k$,

$$\left[\int_\Omega \left(\frac{\partial \phi_h}{\partial t} + (\boldsymbol{u}\cdot\nabla\phi_h)\right)\psi_h\right] + S(\phi_h,\psi_h) = 0 \tag{3}$$

where $S(\phi_h,\psi_h)$ is the GLS stabilization bilinear, which vanishes as $h \to 0$, and $\mathcal{P}_h^k$ is the chosen Galerkin function space, here defined as the discrete (h-dependent) finite element space spanned by Lagrange polynomials of order k.

As already mentioned, the advection of the level set does not preserve the "distance" (i.e. $|\nabla\phi| = 1$) property. This can lead to numerical instabilities due to the accumulation or rarefaction of the level set iso-lines which implicitly provide the smoothed interface-related functions. To overcome this issue, we periodically reset ϕ to a distance function, either solving a discretized Hamilton-Jacobi equation as proposed in [4, 5], or directly reinitializing ϕ with the fast-marching method [6]. In practice, we reset the level set to a signed distance function at fixed rate every ten time steps.

2.3. Fluid equations

The inner and outer fluids are governed by the incompressible Navier-Stokes equations for Newtonian fluids:

$$\rho_\phi\left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\right) - \nabla\cdot\left(\mu_\phi\left[\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T\right]\right) + \nabla p = \boldsymbol{f}, \tag{4}$$

$$\nabla\cdot\boldsymbol{u} = 0, \tag{5}$$

where $\boldsymbol{u}$ is the fluid velocity, p the pressure, $\boldsymbol{f}$ the external forces exerted on the fluid, and ρ_ϕ and μ_ϕ are the level-set-dependant density and viscosity defined in eq. (2). In practice, the right-hand-side force term in eq. (4) accounts for both volumic forces, such as gravitation, and surface forces, such as the surface tension.

The problem definition eqs. (4) and (5) is completed with Dirichlet- or Neumann-type boundary conditions.

2.4. Coupling of the fluid and level set equations

The fluid equations are then coupled with the level set advection equation explicitely following a non-monolithic approach. At each time step, the fluid equations are first solved with the interface-related quantities and surface forces computed using the last-step level set function. We then use the obtained fluid velocity to advect the level set and get the new interface position.

Note that the successive resolution of the fluid and level set equations can also be iterated within one time step, until a fix point of the system of equations is reached. In practice however, for reasonably small time steps, the fix-point solution is already obtained after the first iteration.

3. 3D simulation setup

We now present a 3D benchmark of our numerical approach, using the Navier-Stokes solver developed with the `Feel++` library described in [17]. This benchmark is a three-dimensional extension of the 2D benchmark introduced in [18] and realised using `Feel++` in [19]. The setup for this benchmark was also used in [20] to compare several flow solvers.

3.1. Benchmark problem

The benchmark consists in simulating the rise of a 3D bubble in a Newtonian fluid. The equations solved are the aforementioned incompressible Navier Stokes equations for the fluid and advection equation for the level set, namely

$$\rho_\phi \left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} \right) - \nabla \cdot \left(\mu_\phi \left[\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T \right] \right) + \nabla p = \boldsymbol{f}_g + \boldsymbol{f}_{st}, \tag{6}$$

$$\nabla \cdot \boldsymbol{u} = 0, \tag{7}$$

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla \phi = 0, \tag{8}$$

where ρ_ϕ and μ_ϕ are the interface-dependent fluid parameters defined above, and $\boldsymbol{f}_g$ and $\boldsymbol{f}_{st}$ are respectively the gravitational and surface tension forces, defined as:

$$\boldsymbol{f}_g = \rho_\phi \, \boldsymbol{g} \tag{9}$$

$$\boldsymbol{f}_{st} = \sigma \kappa \boldsymbol{n} \Big|_\Gamma \simeq \sigma \kappa \boldsymbol{n} \, \delta_\varepsilon(\phi) \tag{10}$$

with $\boldsymbol{g} \equiv -0.98 \, \boldsymbol{e}_z$ the gravity acceleration, σ the surface tension, $\boldsymbol{n} = \frac{\nabla \phi}{|\nabla \phi|}$ the normal to the interface and $\kappa = \nabla \cdot \boldsymbol{n}$ its curvature. Equations (6) to (8) are solved after discretization of their variational formulation as presented in section 2.

We consider Ω a cylinder with radius $R = 0.5$ and height $H = 2$, filled with a fluid and containing a droplet of another imiscible fluid. We denote $\Omega_1 = \{\boldsymbol{x} | \phi(\boldsymbol{x}) > 0\}$ the domain outside the droplet, $\Omega_2 = \{\boldsymbol{x} | \phi(\boldsymbol{x}) < 0\}$ the domain inside the bubble and $\Gamma = \{\boldsymbol{x} | \phi(\boldsymbol{x}) = 0\}$ the interface. We impose no-slip boundary conditions $\boldsymbol{u}\big|_{\partial\Omega} = 0$ on Ω walls. The simulation is run from $t = 0$ to 3.

Initially, the bubble is spherical with radius $r_0 = 0.25$ and is centered on the point $(0.5, 0.5, 0.5)$ assuming that the bottom disk of the Ω cylinder is centered at the origin. Figure 1 shows this initial setup.

We denote with indices 1 and 2 the quantities relative to the fluid in respectively Ω_1 and Ω_2. The parameters of the benchmark are then ρ_1, ρ_2, μ_1, μ_2 and σ. We also define two dimensionless numbers to characterize the flow: the Reynolds number is the ratio between inertial and viscous terms and is defined as

$$\mathrm{Re} = \frac{\rho_1 \sqrt{|\boldsymbol{g}|(2r_0)^3}}{\nu_1},$$

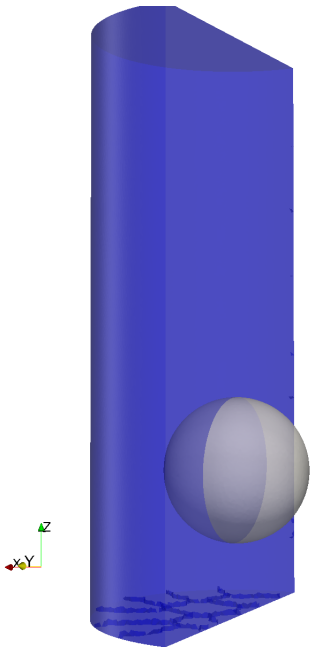


Figure 1: Initial setup for the benchmark.

and the Eötvös number represents the ratio between the gravity force and the surface tension

$$\mathrm{E}_0 = \frac{4\rho_1 |\boldsymbol{g}| r_0^2}{\sigma}.$$

Table 1 reports the values of the parameters used for two different test cases proposed in [20]. At $t = 3$, the first one leads to an ellipsoidal-shaped drop while the second one gives a skirted shape due to the larger density and viscosity contrasts between the inner and outer fluids.

Tests	ρ_1	ρ_2	ν_1	ν_2	σ	Re	E_0
Case 1 (ellipsoidal drop)	1000	100	10	1	24.5	35	10
Case 2 (skirted drop)	1000	1	10	0.1	1.96	35	125

Table 1: Numerical parameters taken for the benchmarks.

To quantify our simulation results, we use three quantities characterizing the shape of the drop at each time-step: the center-of-mass

$$\boldsymbol{x_c} = \frac{1}{|\Omega_2|} \int_{\Omega_2} \boldsymbol{x},$$

the rising velocity – focusing on the vertical component

$$\boldsymbol{u_c} = \frac{1}{|\Omega_2|} \int_{\Omega_2} \boldsymbol{u},$$

and the sphericity – defined as the ratio between the area of a sphere with same volume and the area of the drop –

$$\Psi = \frac{4\pi \left(\frac{3}{4\pi} |\Omega_2| \right)^{\frac{2}{3}}}{|\Gamma|}$$

of the drop. Note that in the previous formulae, we have used the usual "mass" and area of the drop, respectively defined as $|\Omega_2| = \int_{\Omega_2} 1$ and $|\Gamma| = \int_{\Gamma} 1$.

3.2. Simulation setup

The simulations have been performed on the supercomputer of the Grenoble CIMENT HPC center up to 192 processors. To control the convergence of our numerical schemes, the simulations have been run with several unstructured meshes, which characteristics are summarized in table 2.

We run the simulations looking for solutions in finite element spaces spanned by Lagrange polynomials of order $(2, 1, 1)$ for respectively the velocity, the pressure and the level set. The corresponding numbers of degrees of freedom for each mesh size are also reported in table 2.

Mesh properties			Finite-element DOF		
h	Tetrahedra	Points	Order 1 DOF	Order 2 DOF	#DOF
0.025	380 125	62 546	62 546	490 300	1 595 992
0.02	842 865	136 932	136 932	1 092 644	3 551 796
0.0175	1 148 581	186 136	186 136	1 489 729	4 841 459
0.015	1 858 603	299 595	299 595	2 415 170	7 844 700
0.0125	2 983 291	479 167	479 167	3 881 639	12 603 251

Table 2: Mesh properties and degrees of freedom: mesh characteristic size, number of tetrahedra, number of points, number of order 1 degrees of freedom, number of order 2 degrees of freedom and total number of degrees of freedom of the simulation.

Numerical parameters			Total time (h)	
h	#proc	Δt	Case 1	Case 2
0.025	64	1×10^{-2}	3.5	3.6
0.02	128	9×10^{-3}	4.8	5.1
0.0175	128	8×10^{-3}	8.9	9.5
0.015	192	7×10^{-3}	12.3	13.5
0.0125	192	6×10^{-3}	33.8	39.6

Table 3: Numerical parameters used for simulations and resulting simulation times for each test case.

The Navier-Stokes equations are linearized using Newton's method and the resulting linear system is solved with a preconditioned flexible Krylov GMRES method using the SIMPLE preconditioner introduced in [21]. The "inversions" of the velocity and pressure block matrices required by the preconditioning are performed using a block Jacobi and an algebraic multigrid (GAMG) preconditioner respectively.

The linear advection equation is solved with a Krylov GMRES method, preconditioned with an Additive Schwarz Method (GASM) using a direct LU method as sub-preconditionner. Most of the results shown below are obtained using the Fast-Marching method to reinitialize the level set function every 10 time-steps. We however also present a comparison between the Fast-Marching and Hamilton-Jacobi reinitialization methods for the finest mesh in each case to ensure consistency of the results.

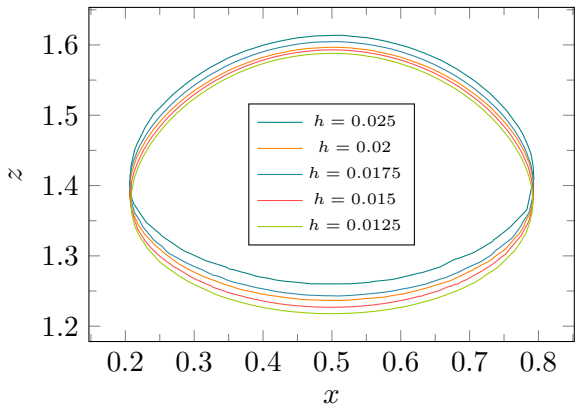
4. Results

4.1. Case 1: the ellipsoidal drop

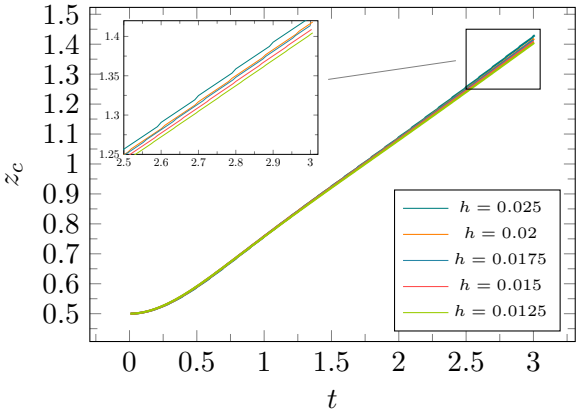
Figure 2a shows the shape of the drop in the $x - z$ plane at the final $t = 3$ time step for the different aforementioned mesh sizes. The shapes are similar and seem to converge when the mesh size is decreasing. The drop reaches a stationary circularity as shown in fig. 2d, and its topology does not change. The velocity increases until it attains a constant value. Figure 2c shows the results obtained for the different mesh sizes. The evolution of the mass of the drop versus time is shown in fig. 2e. It highlights the rather good mass conservation property of our simulation setup, as about 3% of the mass

is at most lost for the coarsest mesh, while the finest one succeeds in keeping the loss in mass below 0.7%.

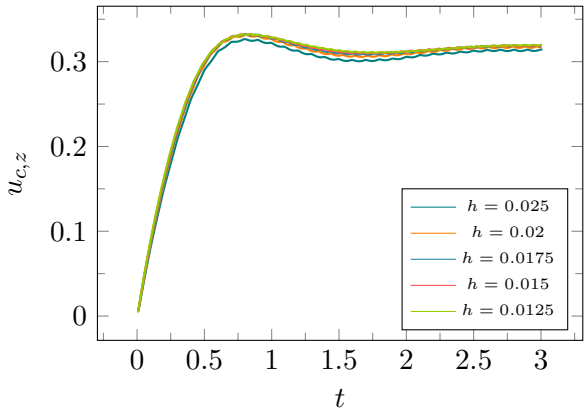
We also note that our simulation perfectly respects the symmetry of the problem and results in a axially symmetric final shape of the drop, as shown in fig. 3.



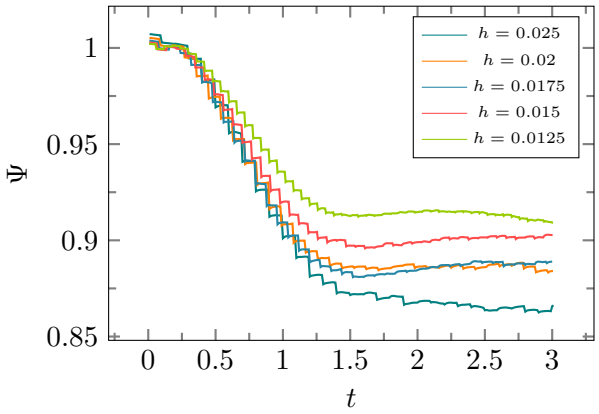
(a) Shape at final time ($t = 3$) in the vertical $x - z$ plane.



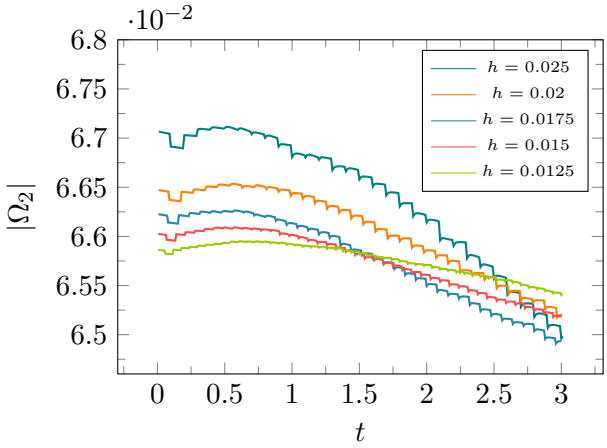
(b) z_c center-of-mass vertical component.



(c) Vertical velocity.



(d) Sphericity.



(e) Mass.

Figure 2: Results for the ellipsoidal test case (case 1).

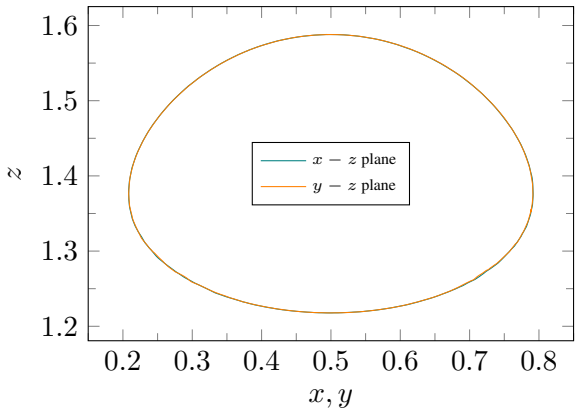


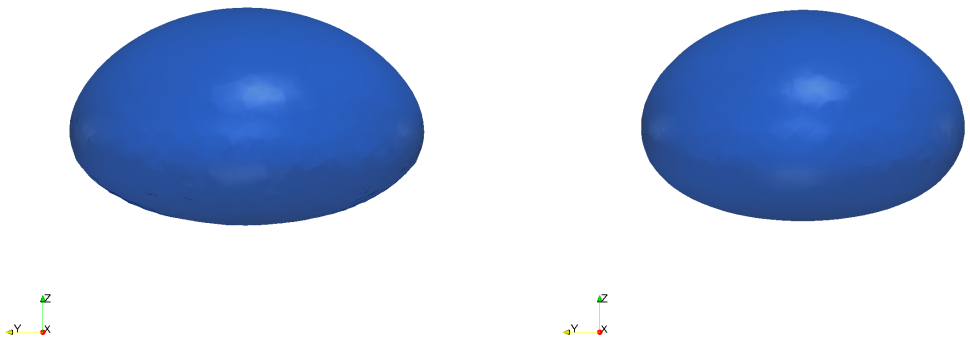
Figure 3: Shape at final time in the $x - z$ and $y - z$ planes for test case 1 ($h = 0.0125$).

4.1.1. Comparison between Hamilton-Jacobi and fast-marching reinitialization

As mentioned in section 2.2, two reinitialization procedures can be used to overcome the "deformation" of the level set which becomes more and more different from the distance to the interface function as it is advected with the fluid velocity. The fast-marching method resets the values of ϕ on the degrees of freedom away from the interface to match the corresponding distance. The Hamilton-Jacobi method consists in solving a advection equation which steady solution is the wanted distance function.

We have run the $h = 0.0175$ simulation with both reinitialization methods to evaluate the properties of each one, and compare them using the monitored quantities. Figure 5 gives the obtained results.

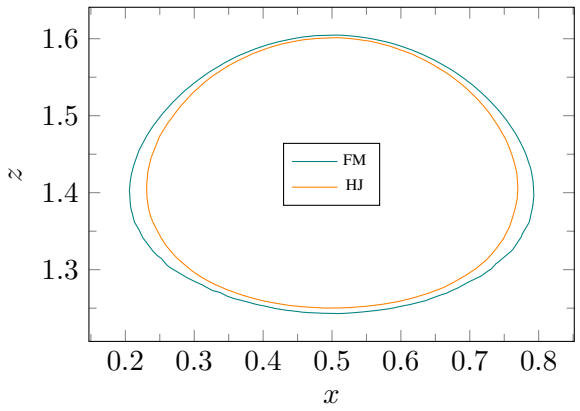
The first observation is that the mass loss (see fig. 5e) is considerably reduced when using the FM method. It goes from about 18% mass lost between $t = 0$ and $t = 3$ for the Hamilton-Jacobi method to less than 2% for the fast-marching method. This resulting difference of size can be noticed in fig. 4. The other main difference is the sphericity of the drop. Figure 5d shows that when using the fast-marching method, the sphericity decreases really quickly and stabilises to a much lower value than the one obtained the Hamilton-Jacobi method. This difference can be explained by the fact that the fast-marching method does not smooth the interface. The shape can then contain some small irregularities leading to a bad sphericity. Evenso, with both methods the sphericity stays quite constant after the first second of the simulation. The rising velocity and the vertical position do not show any significant difference between the two reinitialization methods.



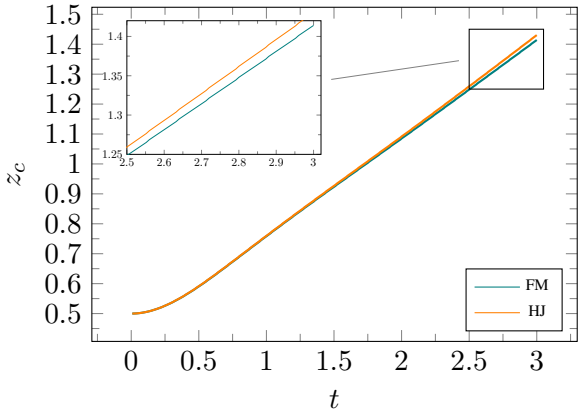
(a) Fast Marching method

(b) Hamilton-Jacobi method

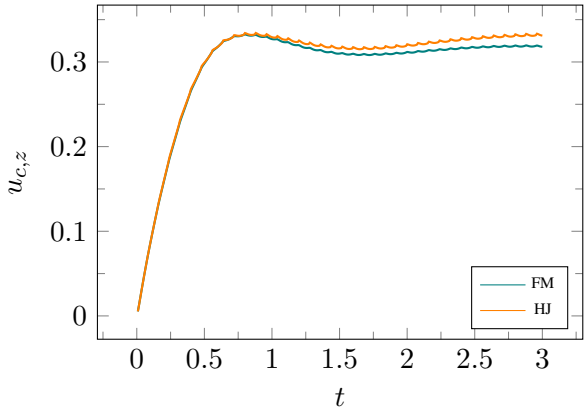
Figure 4: 3D shape at final time ($t = 3$) in the $x - y$ plane for test case 1 ($h = 0.0175$).



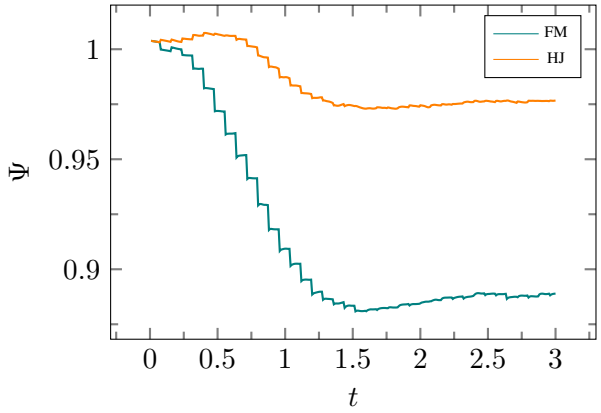
(a) Shape at final time ($t = 3$) in the vertical $x - z$ plane.



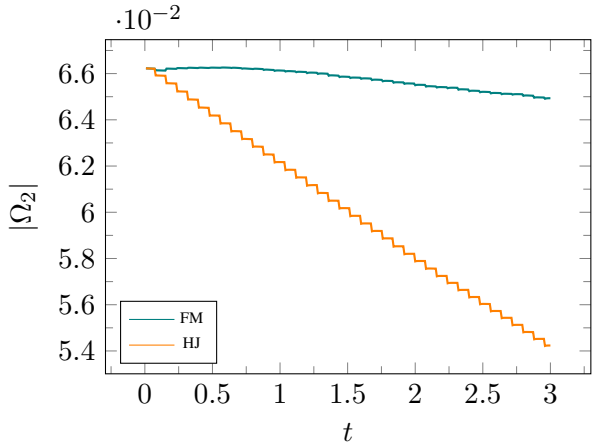
(b) z_c center-of-mass vertical component.



(c) Vertical velocity.



(d) Sphericity.



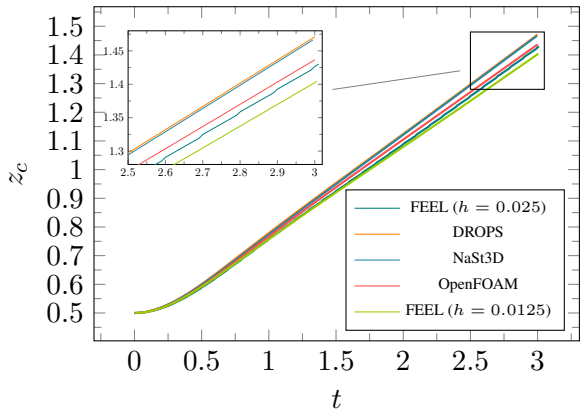
(e) Mass.

Figure 5: Comparison between the Fast Marching method (FM) and the Hamilton-Jacobi (HJ) method for test case 1 (ellipsoidal drop). The characteristic mesh size is $h = 0.0175$.

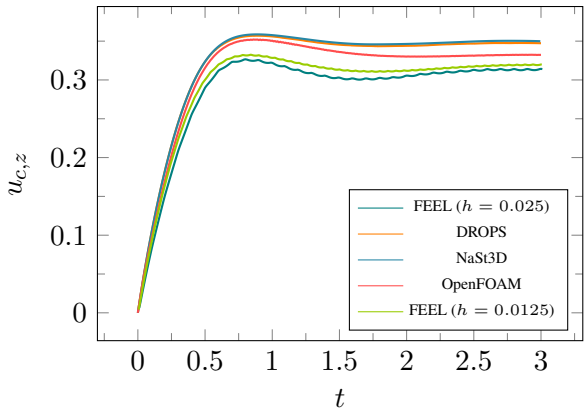
4.1.2. *Comparison with previous results*

Figure 6 shows a plot of our results compared to the ones presented in [20]. In this paper, the authors perform simulations on the same setup and with the same test cases as considered here. To ensure consistency of their results, they use three different flow solvers (hence three different space discretization methods) coupled with two different interface capturing methods: the DROPS and NaSt3D solvers coupled to a level set approach, and the OpenFOAM solver which uses a volume-of-fluid method.

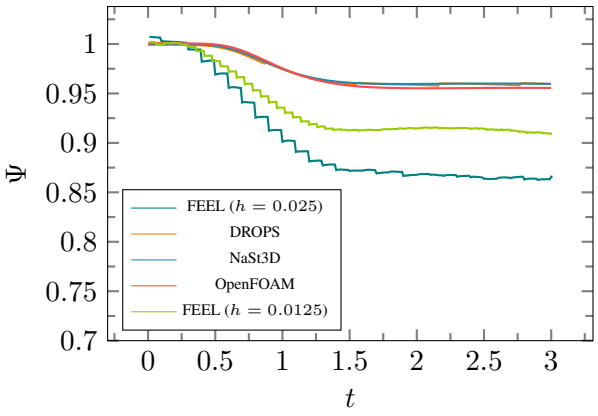
To evaluate the effect of the characteristic mesh size, we plot the results we obtained for the simulations run with both $h = 0.025$ and $h = 0.0125$ along with the results from [20].



(a) z_c center-of-mass vertical component.



(b) Vertical velocity.



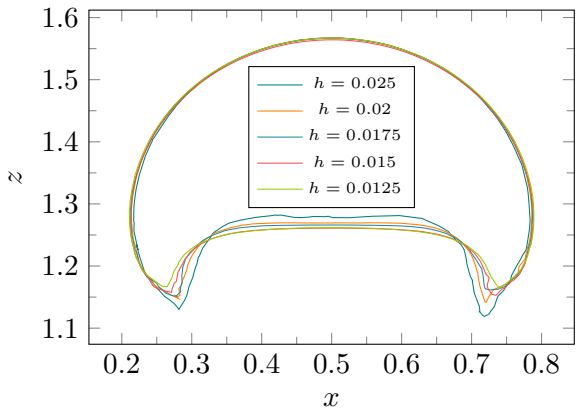
(c) Sphericity.

Figure 6: Comparison between our results (denoted FEEL) and the ones from [20] for the test case 2 (the ellipsoidal drop).

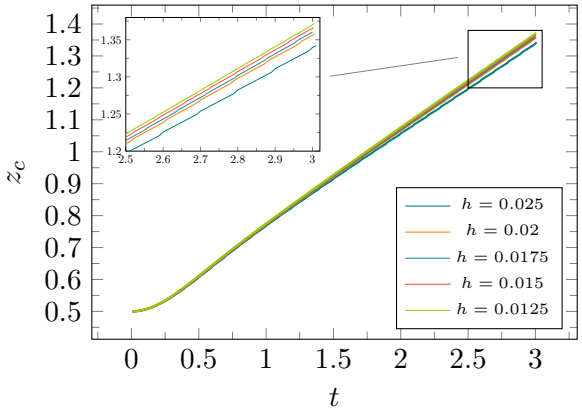
We can observe an overall good agreement between our results and the benchmark performed in [20].

4.2. *Case 2: the skirted drop*

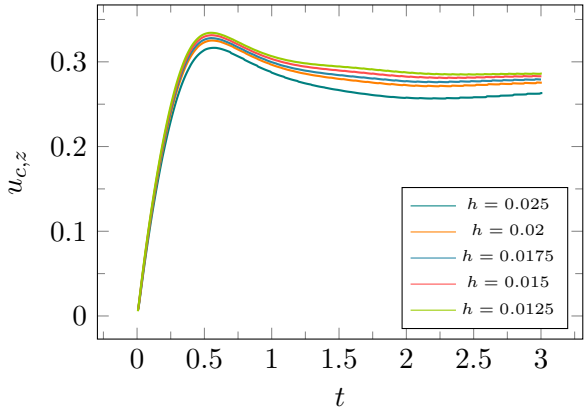
In the second test case, the drop gets more deformed because of the lower surface tension and the higher viscosity and density contrasts. Figure 7 displays the monitored quantities for this test case. We observe that the shape of the "skirt" of the drop at $t = 3$ is quite strongly mesh dependent, but converges as the mesh is refined. The other characteristics of the drop are not so dependent on the mesh refinement, even for the geometrically related ones, such as the drop mass, which shows a really small estimation error (only 2% difference between the coarsest and finest meshes), and displays the really good conservation properties of our simulations. We again also note in fig. 8 the symmetry of the final shape of the drop, which highlights the really good symmetry conservation properties of our approach.



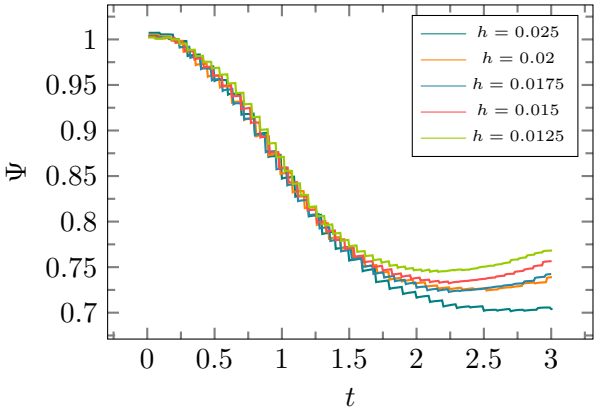
(a) Shape at final time ($t = 3$) in the vertical $x - z$ plane.



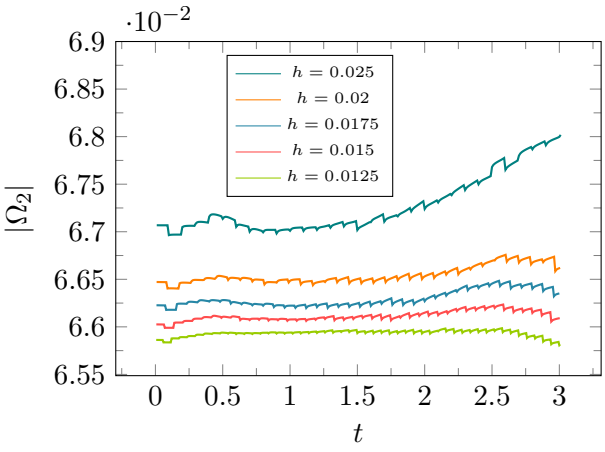
(b) z_c center-of-mass vertical component.



(c) Vertical velocity.



(d) Sphericity.



(e) Mass.

Figure 7: Results for the skirted test case (case 2).

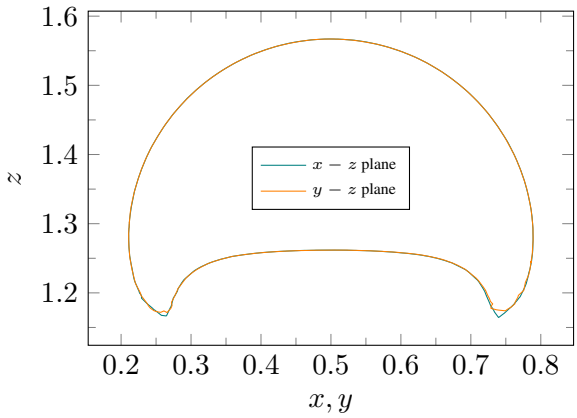
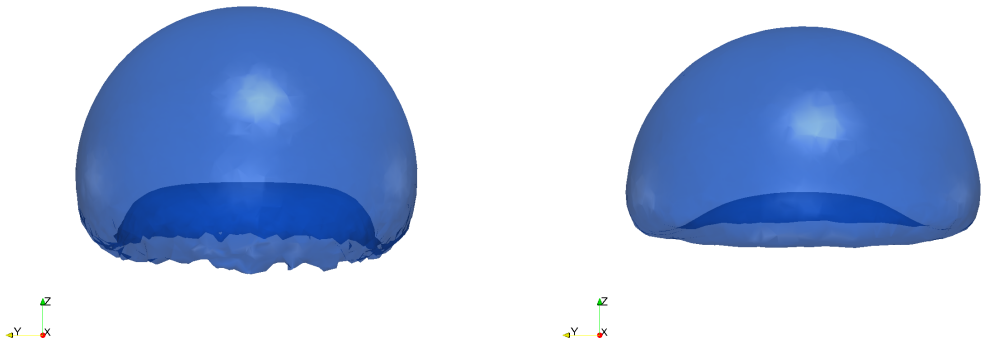


Figure 8: Shape at final time in the $x - z$ and $y - z$ planes for test case 2 ($h = 0.0125$).

4.2.1. Comparison between Hamilton-Jacobi and fast-marching reinitialization

As for the test case 1, we provide a comparison of the results for the test case 2 obtained using either the fast-marching or the Hamilton-Jacobi reinitialization method. These results, obtained for an average mesh size ($h = 0.0175$) are shown in fig. 10. As before, they highlight noticeable differences between the two methods for geometrically related quantities such as mass loss, sphericity and final shape. We can even observe a non-negligible difference for the latter in the region of the "skirt". This difference, mainly related to the diffusive properties of the Hamilton-Jacobi method, can also be observed on the 3D shapes in fig. 9. The good agreement of the results obtained using the fast-marching method tend to suggest that the Hamilton-Jacobi method is not accurate enough for this kind of three-dimensional simulation.



(a) Fast Marching method

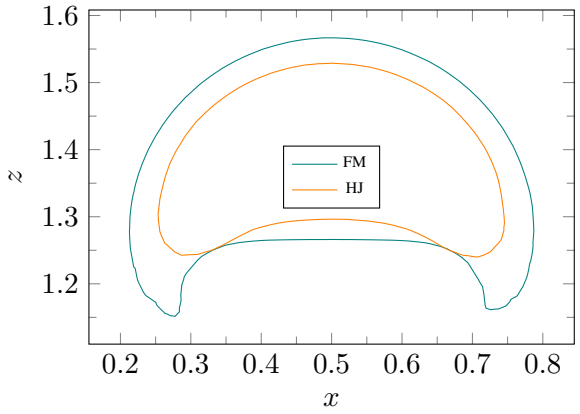
(b) Hamilton-Jacobi method

Figure 9: 3D shape at final time ($t = 3$) in the $x - y$ plane for test case 2 ($h = 0.0175$).

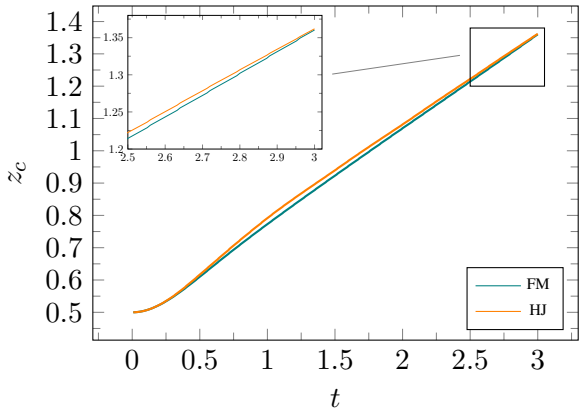
4.2.2. Comparison with previous results

As in section 4.1.2, we compare our results to the benchmark [20], and show the relevant quantities in fig. 11.

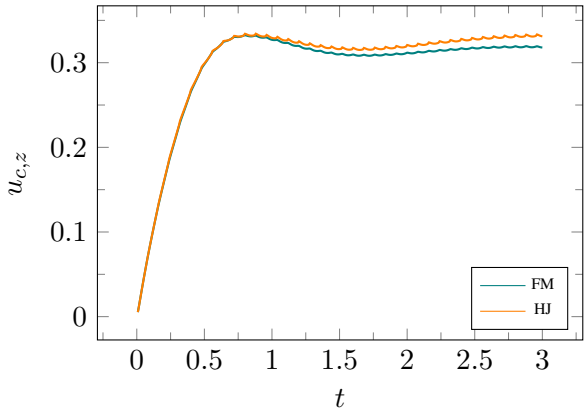
We also observe a good agreement between our simulations and the ones from the benchmark. We however note that the final shape of the skirted drop is very sensitive to the mesh and none of the groups agree on the exact shape which can explain the differences that we see on the parameters in fig. 11 at time $t > 2$.



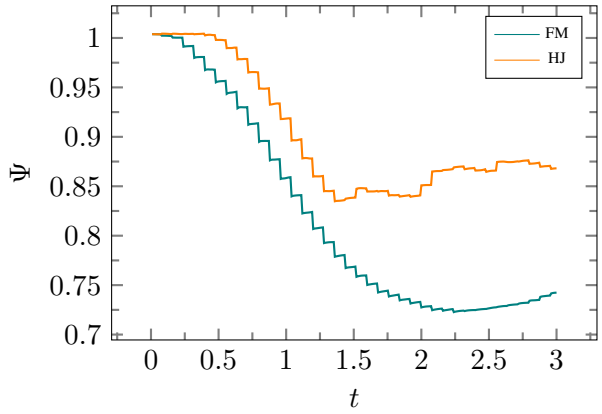
(a) Shape at final time ($t = 3$) in the vertical $x - z$ plane.



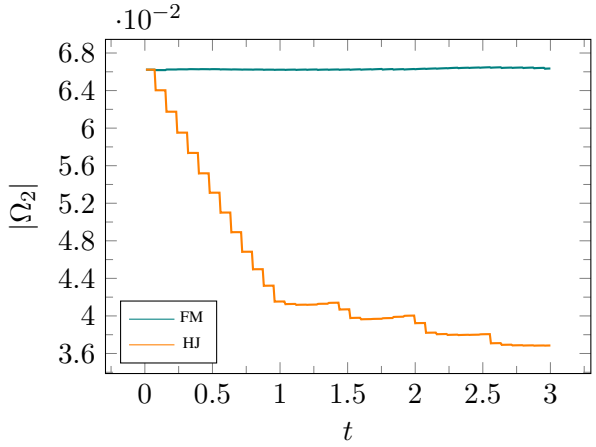
(b) z_c center-of-mass vertical component.



(c) Vertical velocity.

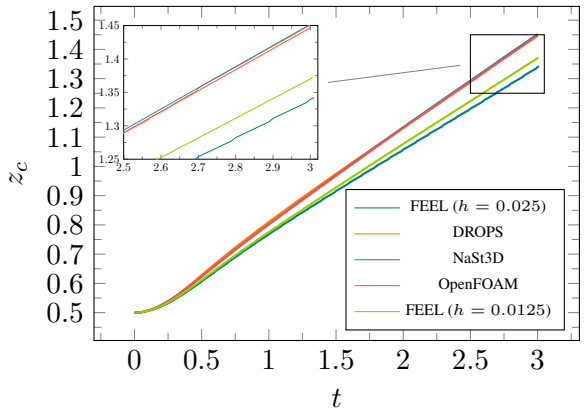


(d) Sphericity.

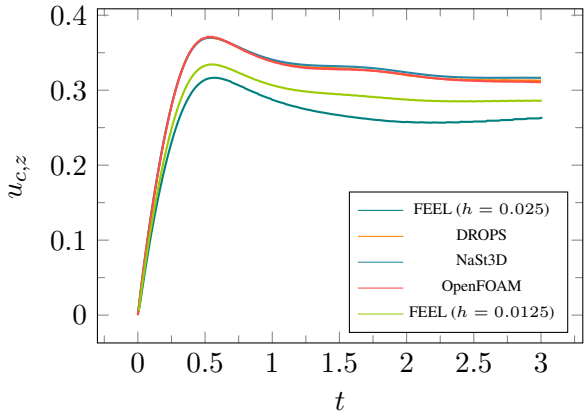


(e) Mass.

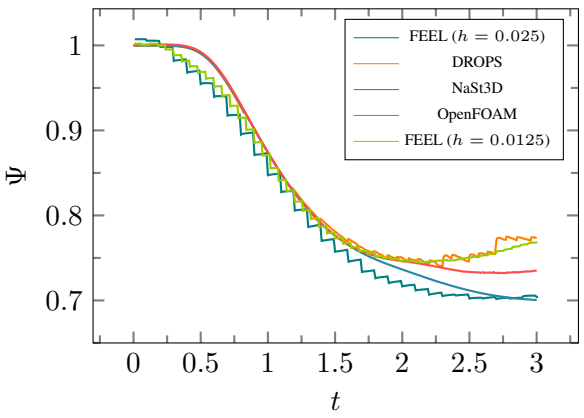
Figure 10: Comparison between the Fast Marching method (FM) and the Hamilton-Jacobi (HJ) method for test case 2 (skirted drop). The characteristic mesh size is $h = 0.0175$.



(a) z_c center-of-mass vertical component.



(b) Vertical velocity.



(c) Sphericity.

Figure 11: Comparison between our results (denoted FEEL) and the ones from [20] for the test case 2 (the skirted drop).

4.3. High-order simulations

As already mentioned, our framework naturally allows the use of high-order Galerkin discretization spaces. As an illustration, we present here benchmark simulation results performed using finite element spaces spanned by Lagrange polynomials of order $(2, 1, 2)$ for each test case. The mesh size considered here is $h = 0.02$, and the results are shown in fig. 12 and fig. 13 for test cases 1 and 2 respectively. We expect the increase in order of the level-set field to improve the overall accuracy.

We can indeed observe that the final shapes of high-order simulations look smoother in both cases, as confirmed by the sphericity plots. The effect is highly noticeable on the "skirt" which appears for the second test-case, which looks even smoother than the one obtained with the finest ($h = 0.0125$) $(2, 1, 1)$ simulation.

We can also notice that more "physically" controlled quantities, such as the position of the center-of-mass and the vertical velocity are less impacted by the polynomial order of the level-set component, which is not so surprising, as these quantities are mainly determined by the (level-set-dependent) fluid equations, which discretization orders where kept constant for this analysis.

Conclusion

We have presented a new numerical framework for the simulation of 3D drops under flow. This framework is based on level-set methods solved by a (possibly high order) finite element method. The explicit coupling between the level-set and the fluid has proven to be efficient and has allowed us to take advantage of reliable and efficient preconditioning strategies to solve the fluid equations. The level-set framework for three-dimensional two-fluid flows has been verified using a standard numerical benchmark and the results are in agreement with the simulations performed with other methods. We have also compared two different level-set reinitialization procedures (the fast-marching and the Hamilton-Jacobi methods) and observed significantlty different behaviors, in particular the former is much better at mass conservation than the latter.

Further improving the accuracy of the level-set and related quantities (such as $\nabla\phi$ or physical quantities defined with ϕ) using higher order and/or hybrid methods is still ongoing.

The framework presented and validated here provides the building blocks for the simulation of complex fluids in complex geometries, and shall be used in a near future to better understand the flow of red blood cells in realistic vascular systems.

Acknowledgments

The authors would like to thank the French National Research Agency (MN-VIVABRAIN projects ANR-12-MONU-0010) for their financial support. Christophe Prud'homme would like to acknowledge the support of the Labex IRMIA. Most of the computations presented in this paper were performed using the Froggy platform of the CIMENT infrastructure https://ciment.ujf-grenoble.fr, which is supported by the Rhône-Alpes region (GRANT CPER07 13 CIRA) and the EquipMeso project (ANR-10-EQPX-29-01).

References

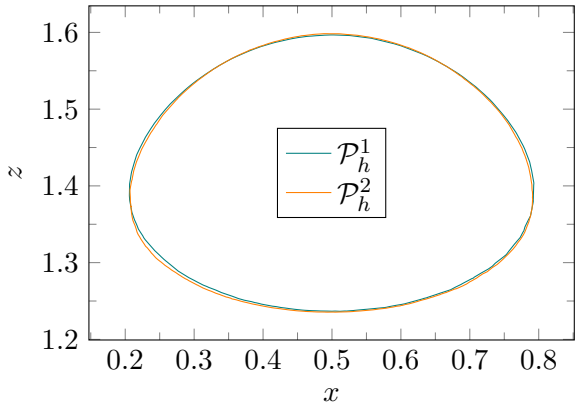
[1] S. Osher, J. A. Sethian, Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations, Journal of computational physics 79 (1) (1988) 12–49.

[2] J. Sethian, Level Set Methods and Fast Marching Methods, Cambridge University Press, 1996.

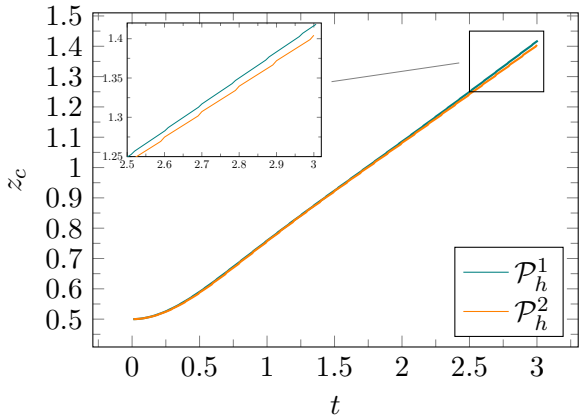
[3] R. F. Stanley Osher, Level Set Methods and Dynamic Implicit Surfaces, Springer, S.S. Antman, J.E. Marsden, L. Sirovich.

[4] E. Rouy, A. Tourin, A viscosity solutions approach to shape-from-shading, SIAM Journal on Numerical Analysis 29 (3) (1992) 867–884.

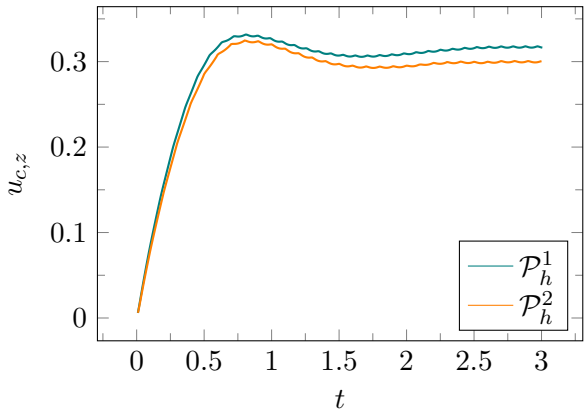
[5] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, Journal of Computational Physics 114 (1) (1994) 146 – 159. doi:http://dx.doi.org/10.1006/jcph.1994.1155.
URL http://www.sciencedirect.com/science/article/pii/S0021999184711557



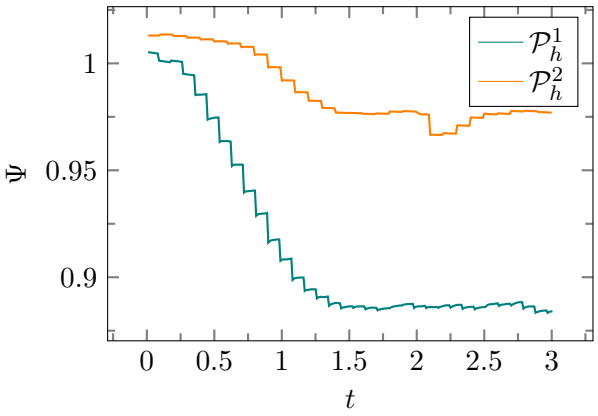
(a) Shape at final time ($t = 3$) in the vertical $x - z$ plane.



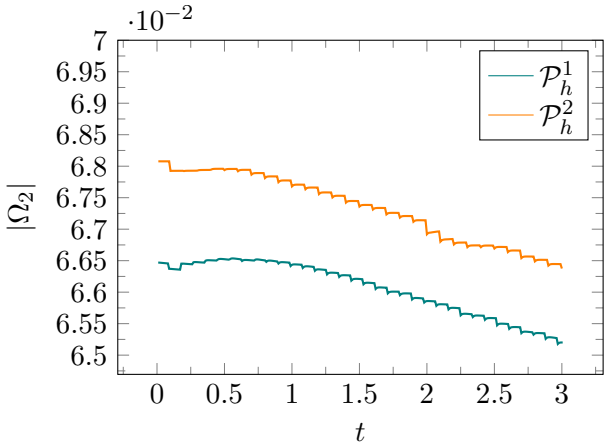
(b) z_c center-of-mass vertical component.



(c) Vertical velocity.

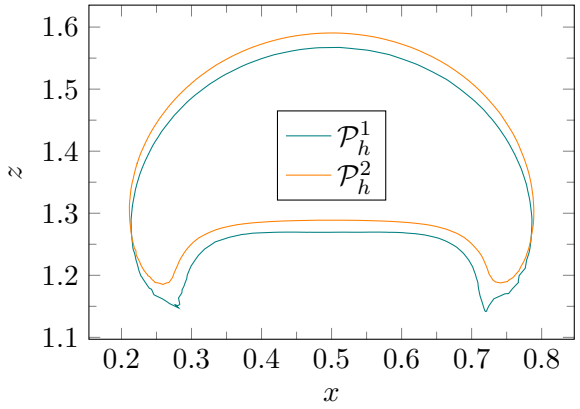


(d) Sphericity.

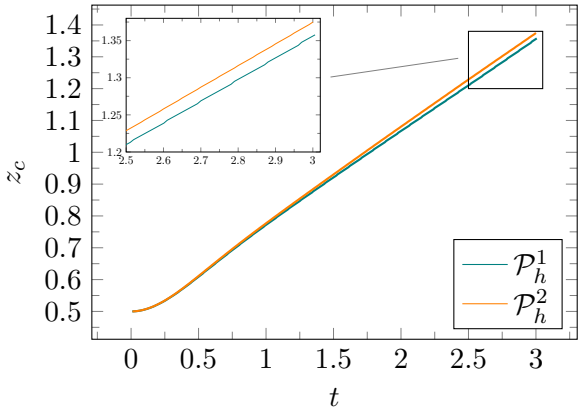


(e) Mass.

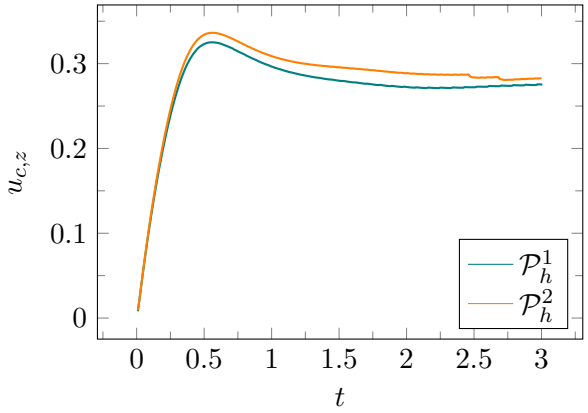
Figure 12: Comparison between $\mathcal{P}_h^1$ and $\mathcal{P}_h^2$ simulations for the ellipsoidal test case ($h = 0.02$).



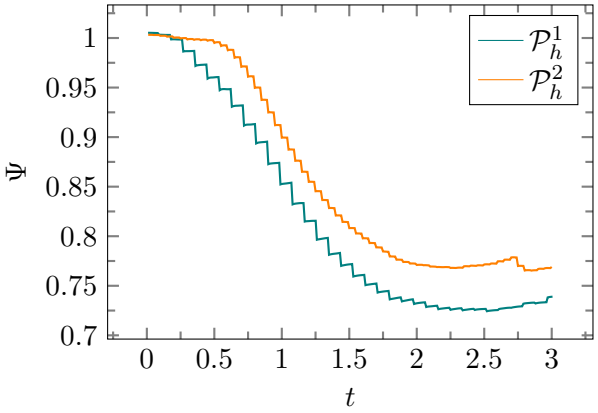
(a) Shape at final time ($t = 3$) in the vertical $x - z$ plane.



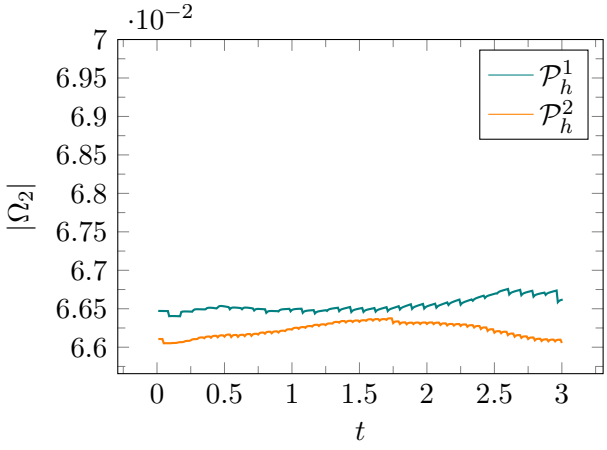
(b) z_c center-of-mass vertical component.



(c) Vertical velocity.



(d) Sphericity.



(e) Mass.

Figure 13: Comparison between $\mathcal{P}_h^1$ and $\mathcal{P}_h^2$ simulations for the skirted test case ($h = 0.02$).

[6] J. A. Sethian, Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science, Vol. 3, Cambridge university press, 1999.
[7] C. Winkelmann, Interior penalty finite element approximation of navier-stokes equations and application to free surface flows, Ph.D. thesis (2007).
[8] E. M. Georges-Henri Cottet, A level set method for fluid-structure interactions with immersed surfaces, Mathematical Models and Methods in Applied Sciences.
[9] C. Prud'Homme, V. Chabannes, V. Doyeux, M. Ismail, A. Samake, G. Pena, Feel++: A Computational Framework for Galerkin Methods and Advanced Numerical Methods (Dec. 2012).
URL http://hal.archives-ouvertes.fr/hal-00662868
[10] C. Prud'homme, V. Chabannes, G. Pena, Feel++: Finite Element Embedded Language in C++, Free Software available at http://www.feelpp.org, contributions from A. Samake, V. Doyeux, M. Ismail and S. Veys.
[11] C. Prud'homme, A domain specific embedded language in C++ for automatic differentiation, projection, integration and variational formulations, Scientific Programming 14.
[12] C. Caldini Queiros, V. Chabannes, M. Ismail, G. Pena, C. Prud'Homme, M. Szopos, R. Tarabay, Towards large-scale three-dimensional blood flow simulations in realistic geometriesPp. 17, Accepted in ESAIM: Proc.
URL http://hal.archives-ouvertes.fr/hal-00786556
[13] V. Chabannes, M. Ismail, C. Prud'Homme, M. SZOPOS, Hemodynamic simulations in the cerebral venous network: A study on the influence of different modeling assumptions, Journal of Coupled Systems and Multiscale Dynamics 3 (1) (2015) 23–37. doi:10.1166/jcsmd.2015.1062.
URL https://hal.archives-ouvertes.fr/hal-01109767
[14] C. Daversin, C. Prudhomme, C. Trophime, Full 3D MultiPhysics Model of High Field PolyHelices Magnets, IEEE Transactions on Applied Superconductivity 26 (4) (2016) 1–4. doi:10.1109/TASC.2016.2516241.
URL https://hal.archives-ouvertes.fr/hal-01220807
[15] L. P. Franca, T. J. Hughes, Two classes of mixed finite element methods, Computer Methods in Applied Mechanics and Engineering 69 (1) (1988) 89 – 129. doi:http://dx.doi.org/10.1016/0045-7825(88)90168-5.
URL http://www.sciencedirect.com/science/article/pii/0045782588901685
[16] T. J. Hughes, L. P. Franca, G. M. Hulbert, A new finite element formulation for computational fluid dynamics: Viii. the galerkin/least-squares method for advective-diffusive equations, Computer Methods in Applied Mechanics and Engineering 73 (2) (1989) 173 – 189. doi:http://dx.doi.org/10.1016/0045-7825(89)90111-4.
URL http://www.sciencedirect.com/science/article/pii/0045782589901114
[17] V. Chabannes, G. Pena, C.Prud'homme, High order fluid structure interaction in 2d and 3d. application to blood flow in arteries, in: Fifth International Conference on Advanced COmputational Methods in ENgineering (ACOMEN 2011), 2011.
[18] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, L. Tobiska, Quantitative benchmark computations of two-dimensional bubble dynamics, International Journal for Numerical Methods in Fluids 60 (11) (2009) 1259–1288. doi:10.1002/fld.1934.
[19] V. Doyeux, Y. Guyot, V. Chabannes, C. PrudHomme, M. Ismail, Simulation of two-fluid flows using a finite element/level set method. application to bubbles and vesicle dynamics, Journal of Computational and Applied Mathematics 246 (2013) 251–259.
[20] J. Adelsberger, P. Esser, M. Griebel, S. Groß, M. Klitz, A. Rüttgers, 3d incompressible two-phase flow benchmark computations for rising droplets, in: Proceedings of the 11th World Congress on Computational Mechanics (WCCM XI), Barcelona, Spain, Vol. 179, 2014.
[21] S. V. Patankar, D. B. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, International journal of heat and mass transfer 15 (10) (1972) 1787–1806.