XL CILAMCE
IBERO-LATIN AMERICAN CONGRESS ON COMPUTATIONAL METHODS IN ENGINEERING
NOVEMBER 11-14, 2019
Praiamar Natal Hotel & Convention
Natal, RN-BRAZIL

IMPLEMENTATION OF THE CONVECTED LEVEL-SET METHOD WITH ADAPTIVE MESH REFINEMENT USING LIBMESH

Malú Grave
malugrave@nacad.ufrj.br
High Performance Computing Center, Federal University of Rio de Janeiro
P.O. Box 68506, RJ 21945-970, Rio de Janeiro, Brazil
José J. Camata
camata@ice.ufjf.br
Department of Computer Science, Federal University of Juiz de Fora
Campus Universidade Federal de Juiz de Fora, Via Local, 4569, MG 36036-900, Juiz de Fora, Brazil
Alvaro L.G.A. Coutinho
alvaro@nacad.ufrj.br
High Performance Computing Center, Federal University of Rio de Janeiro
P.O. Box 68506, RJ 21945-970, Rio de Janeiro, Brazil

Abstract. In the computation of flow problems, the modeling of moving boundaries and interfaces is still challenging. There are different techniques to model such problems: interface-tracking and interface-capturing methods. Here we study an interface-capturing method called convected level-set. The difference to the standard level-set method is that the reinitialization step is embedded in the transport equation model, avoiding a separate step during the calculation. We also use a hyperbolic tangent distance function to get a smooth truncation far from the interface. We couple this interface-capturing method with the Navier-Stokes equations to simulate free-surface problems. We implement all equations on `libMesh`, an open finite element library that provides a framework for multiphysics, considering adaptive mesh refinement. Numerical results are presented for classical interface-capturing benchmarks to study the influence of mesh refinement, time-step, and time discretization. Finally, we present the results of a free-surface problem, simulating a dam-break benchmark.

Keywords: Interface-capturing, Level-Set, Finite Elements, Two-phase Flows

1 Introduction

One challenging category of problems in flow computation is the modeling of moving boundaries and interfaces. These include fluid-particle, fluid-object and fluid-structure interactions, free-surface and two-fluid flows, and flows with moving mechanical components. There are two techniques to model such problems: interface-tracking and interface-capturing (see Bazilevs et al. [1]). An interface-tracking technique requires meshes that "track" the interfaces and are updated as the flow evolves. In other words, interface-tracking is Lagrangian. On the other hand, in interface-capturing techniques, the computations are based on fixed (Eulerian) spatial domains, where an interface function, marking the location of the interface, is computed to "capture" the interface.

Interface-capturing techniques are more flexible than interface-tracking techniques because they do not require mesh updating. They require little effort to represent all complicated features of moving interfaces, and their implementation and post-processing are straightforward. The main drawback of interface-capturing methods is the need to average the fluid properties at the interface elements due to the discontinuity of the Eulerian representation of the interface. A well-known interface-capturing method is the Volume of Fluid method (VOF). It was first introduced by Hirt and Nichols [2] and employs a step function equal to unity at any point occupied by one phase and zero at points occupied by the other one. In this sense, the interface is implicitly represented by the partially filled elements. However, it is challenging to advect a discontinuous step function. Thus, other techniques, as level-sets, appear to heal this situation.

The level-set method was first introduced by Osher and Sethian [3] in the late 1980s as a technique for capturing evolving interfaces and tracking the propagation of fronts. The method has been used successfully in many fluid flow applications (Sussman et al. [4], Osher and Sethian [3], Gibou et al. [5], Ville et al. [6], Bonito and Guermond [7]), but it can also be used in other contexts like shape optimization as in Dapogny et al. [8], Yamada et al. [9]. This method consists in to separate two phases with signed distance functions (SDFs), in which the interface between phases is defined as the SDF zero level-set. The level-set function determines the smallest distance of a point to the interface, and the signal is used to represent the different phases (positives values represent one phase, while negatives represent the other one). Eq. (1) defines the level-set function ϕ.

$$\phi(\mathbf{x}) = \begin{cases} d(\mathbf{x},\Gamma) \text{ for } \mathbf{x} \in \Omega^{+} \\ 0 \text{ for } \mathbf{x} \in \Gamma \\ -d(\mathbf{x},\Gamma) \text{ for } \mathbf{x} \in \Omega^{-} \end{cases} \tag{1}$$

where $d(\cdot,\cdot)$ is the Euclidian distance to Γ, Ω^{+} and Ω^{-} are the subdomains corresponded to each phase and Γ is the interface between the subdomains.

Note that the gradient of the level-set function remains constant along the domain, i.e. this SDF satisfies the Eikonal equation $||\nabla\phi||= 1$. However, this property is not preserved in the numerical process of advection. In some regions, the level-set function can become very steep, so that a nearly discontinuous function has to be advected. In other regions, it may become very flat, causing a significant loss of accuracy when localizing the interface. Therefore, to preserve the SDF property, reinitialization procedures are commonly applied.

The level-set method can be generalized to any SDF having the same independent measure properties. The only useful information for our computation is the exact zero isovalue position. Thus, the level-set function does not need to be applied in all domain. Furthermore, this may be the cause of numerical instabilities. Therefore, we may cut off the level-set function at a thickness E using a sinusoidal filter as presented in Coupez [10] (Eq. 2) or as presented in Khalloufi et al. [11] (Eq. 3).

$$\tilde{\phi} = \begin{cases} \dfrac{2E}{\pi} \text{ for } \phi > E \\ \\ \dfrac{2E}{\pi} sin\left(\dfrac{\pi}{2E}\phi\right) \text{ for } \phi \in [-E, E] \\ \\ -\dfrac{2E}{\pi} \text{ for } \phi < -E \end{cases} \tag{2}$$

$$\tilde{\phi} = \ E \tanh\left(\frac{\phi}{E}\right) \tag{3}$$

Both modified level-set functions $\tilde{\phi}$ gives us a way to define a smooth truncation of the distance function. Moreover, it can be generalized to any number of spatial dimensions. Here, we use Eq. (3), as the modified level-set function. For the sake of simplicity in the notations, we drop in the following the tilde and ϕ denotes the truncated level-set function.

Another characteristic of a SDF is that the normal vector to the interface is readily obtained by

$$\mathbf{n} = \nabla\phi. \tag{4}$$

This is particularly useful when it comes to recovering interface-related quantities such as the curvature $\left(\boldsymbol{\kappa}(\phi) = \nabla \cdot \frac{\nabla\phi}{||\nabla\phi||}\right)$, which may be applied in free surface problems, when modeling the surface tension.

The objective of this study is to use and understand the convected level-set method to compute two-phase problems. The remainder of this paper is organized as follows: the governing equations are presented in section 2. Section 3 outlines the discretization and stabilization schemes used in this study. Details about adaptive mesh refinement are presented in Section 4. Afterwards, in section 5, we present the performance of the numerical method by solving several benchmark problems. Finally, in section 6, a conclusion is given with a discussion and outlook.

2 Governing equations

Assuming that a velocity field, $\mathbf{u}$, is known everywhere in Ω, the interface motion is given by the following transport equation:

$$\begin{gathered} \frac{\partial\phi}{\partial t} + \mathbf{u} \cdot \nabla\phi = 0 \\ \phi(t = 0, \mathbf{x}) = \phi_0(\mathbf{x}). \end{gathered} \tag{5}$$

The equation above does not preserve any geometric property of the level-set function and solving this equation numerically usually provides a function which is not a SDF. Thus, to avoid numerical instabilities, it is necessary to reinitialize the distance function.

Classically, reinitialization is obtained by solving a Hamilton-Jacobi equation that reconstructs the level-set with the exact zero isovalue of $\phi(\mathbf{x}, t)$. If we introduce a virtual time τ, we search $\beta(\mathbf{x}, \tau)$, that has the same zero values than $\phi(\mathbf{x}, t)$, solving the following equation to keep the slope close to a given function S, that is,

$$\begin{gathered} \frac{\partial\beta}{\partial\tau} + \text{sign}(\phi)(|\nabla\beta| - S) = 0 \\ \beta(\tau = 0, x) = \phi(x, t). \end{gathered} \tag{6}$$

Here $sign(\phi)$ is the sign function defined as:

$$sign(\phi) = \begin{cases} 1 \text{ for } \phi > 0 \\ 0 \text{ for } \phi = 0 \\ -1 \text{ for } \phi < 0 \end{cases} \tag{7}$$

Note that $sign(\phi) = 0$ where $\phi = 0$, thus this reinitialization step does not influence the interface position. We want to keep the slope close to the gradient of the level-set function, that is

$$S = ||\nabla\phi|| = 1 - \left(\frac{\phi}{E}\right)^2 \tag{8}$$

It is possible to avoid the reinitialization step using the called convected reinitialization (Coupez [10], Ville et al. [6]). Once the Hamilton-Jacobi reinitialization is combined to the convection equation, it can be solved using:

$$\begin{aligned} \frac{\partial\phi}{\partial t} + \mathbf{u}\cdot\nabla\phi + \lambda\text{sign}(\phi)\left(|\nabla\phi| - S\right) = 0 \\ \phi(t=0,\mathbf{x}) = \phi_0(\mathbf{x}) \end{aligned} \tag{9}$$

where $\lambda = \frac{h_e}{\Delta t}$ and h_e is the characteristic mesh size.

Now, introducing the convection velocity coming from the Hamilton-Jacobi equation, we can rewrite Eq. (9) as

$$\begin{aligned} \frac{\partial\phi}{\partial t} + (\mathbf{u} + \lambda\mathbf{U})\cdot\nabla\phi - \lambda\text{sign}(\phi)S = 0 \\ \phi(t=0,\mathbf{x}) = \phi_0(\mathbf{x}) \end{aligned} \tag{10}$$

where $\mathbf{U} = \text{sign}(\phi)\dfrac{\nabla\phi}{||\nabla\phi||}$.

Since $\mathbf{U}$ is only defined where $||\nabla\phi|| \neq 0$, it must be regularized, and we do that with the following expression (Coupez [12])

$$\mathbf{U}_\epsilon = \text{sign}(\phi)\frac{\nabla\phi}{\epsilon + (1-\epsilon)||\nabla\phi||} \tag{11}$$

where $\epsilon \approx h_e$. Consequently, the slope S must also be slightly modified.

$$S_\epsilon = \frac{||\nabla\phi||}{\epsilon + (1-\epsilon)||\nabla\phi||}S \tag{12}$$

Finally, the final model is given by

$$\begin{aligned} \frac{\partial\phi}{\partial t} + (\mathbf{u} + \lambda\mathbf{U}_\epsilon)\cdot\nabla\phi - \lambda\text{sign}(\phi)S_\epsilon = 0 \\ \phi(t=0,\mathbf{x}) = \phi_0(\mathbf{x}) \end{aligned} \tag{13}$$

It is important to remember that Eq. (13) is non-linear, because $\mathbf{U}_\epsilon$ depends on ϕ. Thus we linearize it by evaluating $\mathbf{U}_\epsilon$ with the value of ϕ at the previous time step. We also evaluate the slope S_ϵ at the previous time step. The main objective of this equation is to maintain the SDF as smooth and local as possible. Note that without the reinitialization, the level-set equation (5) is similar to the VOF method (Elias and Coutinho [13]).

3 Discretization and stabilization schemes

For time integration of Eq. (13) we use Backward Differentiation Formulas of different orders:

- B-E (Backward-Euler):

$$y^{n+1} - y^n = \Delta t f(t^{n+1}, y^{n+1}) \tag{14}$$

- BDF2 (order two):

$$1.5y^{n+1} - 2y^n + 0.5y^{n-1} = \Delta t f(t^{n+1}, y^{n+1}) \tag{15}$$

- BDF3 (order three):

$$11/6y^{n+1} - 3y^n + 1.5y^{n-1} - 1/3y^{n-2} = \Delta t f(t^{n+1}, y^{n+1}) \tag{16}$$

Here, y^{n+1} is the unknown at $t = t^{n+1}$ and y^n, y^{n-1} and y^{n-2} are evaluated at previous time steps. For the first time steps of BDF2 and BDF3, we use Backward-Euler. Let us assume that we have constructed some suitably-defined finite-dimensional trial solution and test function spaces S^h_ϕ and V^h_ϕ, respectively. The stabilized finite element formulation of Eq. (13) can then be written as follows: find $\phi^h \in S^h_\phi$ such that $\forall w^h \in V^h$:

$$\begin{aligned} &\left(w^h, \frac{\partial \phi^h}{\partial t} + (\mathbf{u}^h + \lambda \mathbf{U}^h_\epsilon) \cdot \nabla \phi^h - \lambda sign(\phi^h) S^h_\epsilon\right) \\ &+ \left((\mathbf{u}^h + \lambda \mathbf{U}^h_\epsilon) \cdot \nabla w^h, \tau_\phi R^h_\phi\right)_{\Omega_e} + (\nabla w^h, \kappa_{dc} \nabla \phi^h) = 0 \end{aligned} \tag{17}$$

where R^h_ϕ is the residual of the transport equation (13), and τ_ϕ is the stabilization parameter of the SUPG formulation (Hughes [14]), that is,

$$R^h_\phi = \frac{\partial \phi^h}{\partial t} + (\mathbf{u}^h + \lambda \mathbf{U}^h_\epsilon) \cdot \nabla \phi^h - \lambda sign(\phi^h) S^h_\epsilon \tag{18}$$

$$\tau_\phi = \frac{1}{D|(\mathbf{u^h} + \lambda \mathbf{U}^h_\epsilon) \cdot \nabla w^h|} \approx \frac{h_e}{D||(\mathbf{u}^h + \lambda \mathbf{U}^h_\epsilon)||} \tag{19}$$

with D being the number of nodes per element,

We also added to the formulation a discontinuity-capturing term κ_{dc}, based on the $YZ\beta$ discontinuity-capturing operator (Bazilevs et al. [15]), defined as

$$\kappa_{dc} = \left|Y^{-1} Z\right| \left(\sum_{i=1}^{n_{dim}} \left|Y^{-1} \frac{\partial \phi^h}{\partial x_i}\right|\right)^{\beta^s/2-1} h_e^{\beta^s} \tag{20}$$

where

$$Y = \bar{\phi} \tag{21}$$

$$Z = R^h_\phi \tag{22}$$

The discontinuity-capturing operator is important when the interface between phases has sharp corners, and it acts smoothing the discontinuity that may appear in these regions. The parameter β^s in Eq. (20) influences the smoothness of the layer (for smoother layers it is set to 1), and the parameter $\bar{\phi}$ is a reference value ($\bar{\phi} = 1$). Note that if $\beta^s = 1$ and the reference value $\bar{\phi} = 1$, the discontinuity-capturing term renders to the Consistent Approximated Upwind method (Galeão and Carmo [16]). The discontinuity-capturing term is non-linear, therefore we linearize it by evaluating κ_{dc} with the value of ϕ^h at the previous Newton iteration.

4 Adaptive Mesh Refinement

All implementations are done using `libMesh`, a C++ FEM open-source software library for parallel adaptive finite element applications (Kirk et al. [17]) which also interfaces with external solver packages like PETSc (Balay et al. [18]) and Trilinos (Heroux et al. [19]). It provides a finite element framework that can be used for the numerical simulation of partial differential equations on serial and parallel platforms. This library is an excellent tool for programming the finite element method and can be used for one-, two-, and three-dimensional steady and transient simulations. The `libMesh` library also has an adaptive mesh refinement (AMR) and coarsening strategy as follows.

The AMR procedure uses a local error estimator considering the error of an element relative to its neighbor elements in the mesh. This error may come from any variable of the system. As it is standard in `libMesh`, Kelly′s error indicator is employed, which uses the H1-seminorm to estimate the error (Ainsworth and Oden [20]). Apart from the element interior residual, the flux jumps across the inter-element faces influence the element error. The flux jump of each face is computed and added to the error contribution of the cell. For both, the residual and flux jump, the value of the desired variables at each node is necessary. Being so, the error $||e||^2$ can be stated as:

$$||e||^2 = \sum_{i=1}^{n} ||e||_i^2 \tag{23}$$

where $||e||_i^2$ is the error of each variable. In this study we use only the velocity as variable for the error estimator.

After computing the error values the elements are "flagged" for refining and coarsening regarding their relative error. This is done by a statistical element flagging strategy. It is assumed that the element error $||e||$ is distributed approximately in a normal probability function. Here, the statistical mean μ_s and standard deviation σ_s of all errors are calculated. Whether an element is flagged is depending on a refining (r_f) and a coarsening (c_f) fraction. For all errors $||e|| < \mu_s - \sigma_s c_f$ the elements are flagged for coarsening and for all $||e|| > \mu_s + \sigma_s r_f$ the elements are marked for refinement. The refinement is performed by local isotropic subdivision (h-refinement) with hanging nodes. Here, the refinement level is limited by a maximum h-level (h_{max}). The coarsening is done by h-restitution of subelements (Peterson [21], Rossa and Coutinho [22], Kelly et al. [23]).

5 Applications

In this section we will assess the performance of the numerical method by solving several benchmark problems.

5.1 Advection of a circle

In this two-dimensional example, in which the side of the computational domain is set to 1 m, we consider the rotation of a circle initially centered at (0.25,0.5) m with a radius of 0.15 m. The imposed velocity field $\mathbf{u}^h$ is:

$$\begin{aligned} u &= -2\pi(y - y_{(Square\,Center)}) \\ v &= 2\pi(x - x_{(Square\,Center)}) \end{aligned} \tag{24}$$

where x and y are the coordinates.

We choose a thickness $E = 0.025$ and we use an adaptive mesh refinement based on the flux jump of the level-set function error, in which $h_{max} = 2$, the coarsening fraction $c_f = 0.01$ and the refining fraction $r_f = 0.99$. We apply the adaptive mesh refinement every 50 time steps. The time discretization used is the BDF2 method. Here we do not use the discontinuity-capturing operator. The initial mesh has

50×50 bilinear elements and after the refinement, the smallest element has 0.005 m of size. We also initially refine in two levels a region where the circle is located. The time step is 0.0005 seconds. Figure 1 shows the simulation results at different steps during one revolution of the circle. The zero level-set is highlighted and it is possible to see the mesh adaptivity.

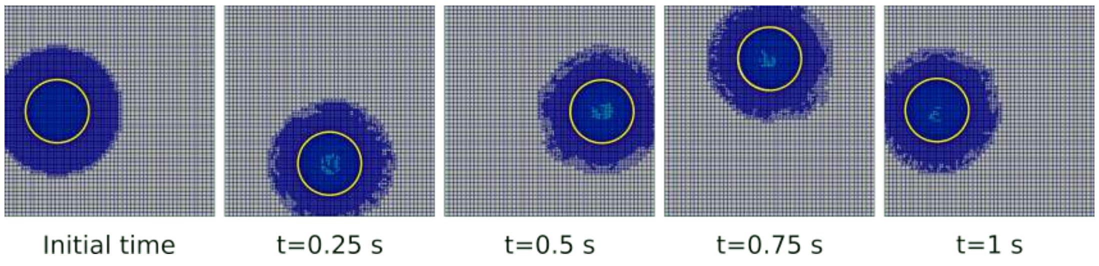


Figure 1. Position of the circle and mesh refinement at different time steps during one turn.

A series of runs of this case with fixed mesh were performed to outline the importance of the mesh size, the corresponding time step, the time discretization method and conservation. Table 1 provides the data used for a selected number of simulations. The computational times were obtained in an Intel$^{©}$ Core^{TM} i5-8400 CPU @ 2.80GHz $\times$ 6. In Figure 2, the shape's deformation between the initial and the final time are shown for all these simulations.

Table 1. Parameters of the different circle rotation simulations performed.

			CPU time (s)	
Case	**Mesh Grid**	**Time Step (s)**	**(B-E)**	**(BDF2)**
(a)	50×50	0.01	10	10
(b)	50×50	0.005	20	17
(c)	50×50	0.0025	30	31
(d)	100×100	0.005	66	66
(e)	100×100	0.0025	117	118
(f)	100×100	0.001	265	266
(g)	200×200	0.001	961	990
(h)	200×200	0.0005	1096	1041

The most challenging feature for a good interface-capturing method resides in its ability to preserve the mass of the species involved. Here the mass conservation is given by the relative error between the final area of the deformed circle and the initial area. Figure 3 shows the percentage of area loss per time of the simulations given by Table 1.

These runs indicate the dependency of the accuracy of the modified level-set method on the mesh size and time discretization method. The mass conservation is also dependent on the discretization of the mesh. Refined meshes loose less mass than the coarse ones. Moreover, refining the domain implies a smaller time step for a better precision. The BDF2 provided more accurate results in all simulations and there is no extra computational cost using it. For BDF3, on coarse meshes, the simulation became unstable and for the refined meshes, the gain on accuracy was minimal. The computational time for the result with adaptive mesh refinement presented in Fig. (1) was 711 seconds and the area loss was 0.07%. Therefore, by using AMR we save computational effort and we obtain the same accuracy of the results of run (h) that took 1096 seconds.

In Figure 4, the isovalues of the modified level-set function of run (h) are plotted and we can compare it when using the VOF method (Elias and Coutinho [13]). In this test case the Backward-Euler and the

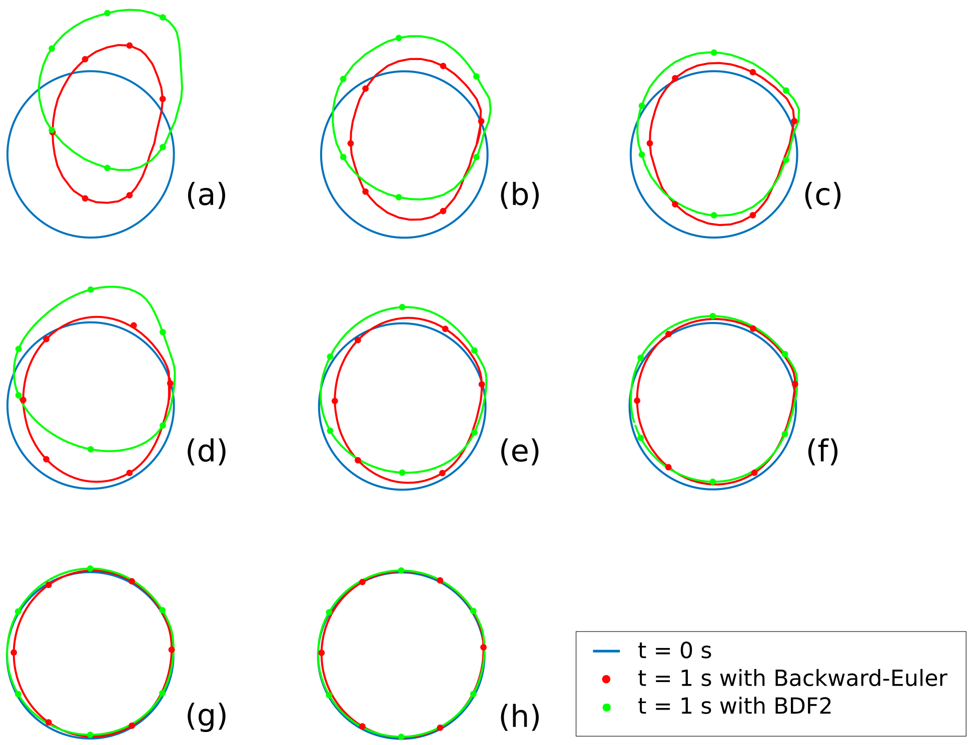


Figure 2. Circle deformation on different meshes and with different time steps. In the simple blue line: the initial zero isovalue of the distance function. In the red line with markers: the same isovalue at the final timestep (after one turn) with the Backward-Euler time discretization. In the green line with markers: the same isovalue at the final timestep (after one turn) with the BDF2 time discretization. (a)-(h) correspond to values given in Table 1.

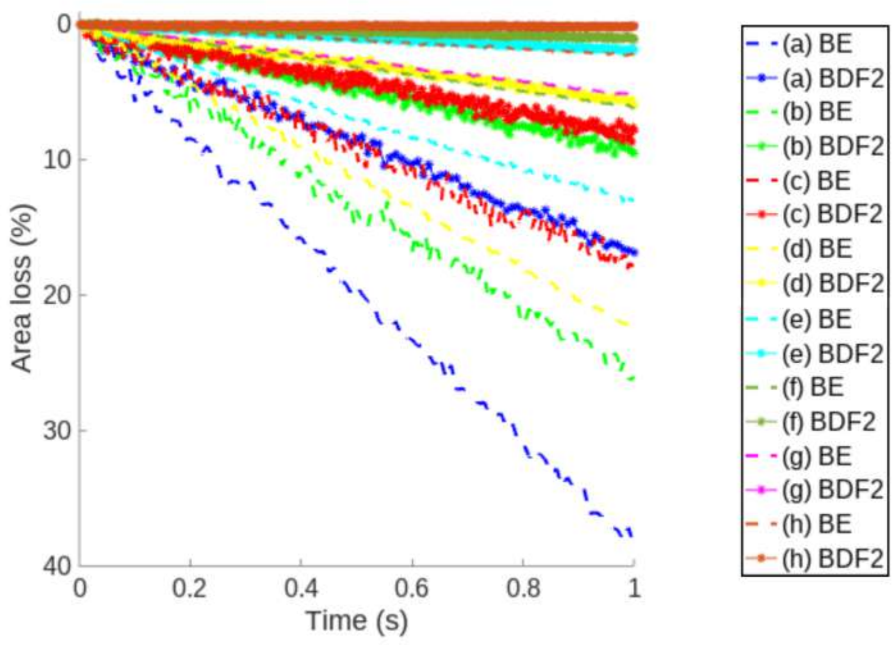
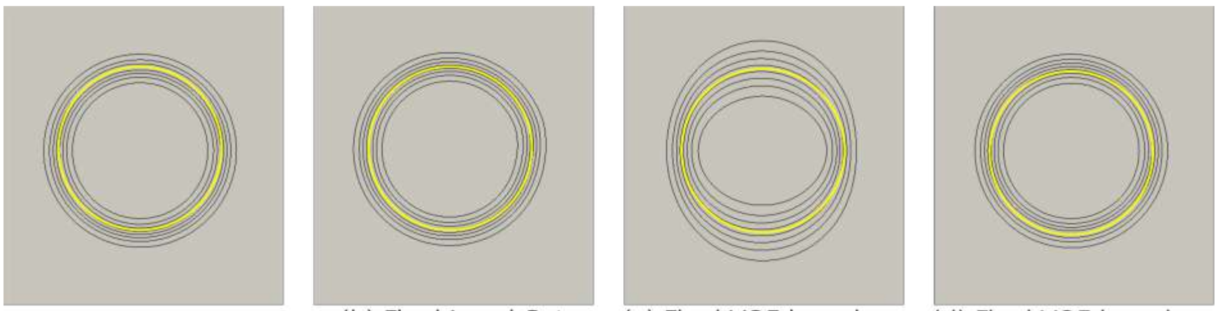


Figure 3. Area loss per time of all simulations [(a)-(h)] given in Table 1.

BDF2 time discretization methods for the modified level-set method provided similar isovalues, thus we are presenting only the isovalues obtained with BDF2 method for simplicity. The isovalues are in the range of -0.025 to 0.025, with spacements of 0.005. In both level-set and VOF methods, we started with the same isovalues, and after one turn we can see that even the zero isovalue for the VOF with the Backward-Euler time discretization method keeps its shape close to a circle, the other isovalues had a distortion and became elypses. On the other hand, the BDF2 method preserve the isovalues shape better. Actually, there is a distortion in the shape of the isovalues, but we cannot see with naked eyes. Therefore, it is possible to see how the reinitialization adjust the level-set field to keep it a signed distance function and consequently to avoid numerical instabilities.



(a) Initial isovalues (b) Final Level-Set isovalues (c) Final VOF isovalues with Backward-Euler (d) Final VOF isovalues with BDF2

Figure 4. Isovalues when using the modified level-set function and the VOF method. (a) Initial isovalues for both methods. (b) Final isovalues using the level-set method with BDF2 (the results using the Backward-Euler are very similar). (c) Final isovalues using the VOF method with Backward-Euler. (d) Final isovalues using the VOF method with BDF2.

5.2 Advection of a sphere

Now we consider the rotation of a sphere initially centered at (0.25,0.5,0.5) m. The side of the computational domain is set to 1 m, with a radius of 0.15 m. The imposed velocity field is:

$$\begin{aligned} u &= -2\pi(y - y_{(Square\,Center)}) \\ v &= 2\pi(x - x_{(Square\,Center)}) \\ w &= 0 \end{aligned} \tag{25}$$

We only use BDF2 for the next simulations. We use an adaptive mesh refinement with $h_{max} = 2$, $c_f = 0.005$ and $r_f = 0.9$. We apply the adaptive mesh refinement every 25 time steps. The initial mesh has $2 \times 25 \times 25$ triliniar hexahedra elements and after the refinement, the smallest element has 0.01 m of side. We also refine two levels of the region where the sphere is initially located, as we can see in Fig. (5). The time step is 0.001 seconds and we choose a thickness $E = 0.025$. Here we add the discontinuity-capturing operator to preclude undesired oscillations. The mass conservation is preserved with a relative error of 2.4%.

Figure 6 shows the simulation at different steps during one rotation. The result obtained shows that the modified level-set method preserves a good interface shape after one turn.

5.3 Zalesak's problem

Zalesak's problem consists mainly in the rotation of a slotted disk under a given velocity field (Zalesak [24]). The side of the computational domain is set to 1.2 m and the disk is initially centered in $(0.3, 0.6)$ with a radius of 0.2 m and a slot length of 0.35 m (Fig. (7)). The velocity field is also given by Eq. (24).

We choose a thickness $E = 0.01$, because a thicker thickness would overlap the isosurfaces due the geometry of the disk. We use an adaptive mesh refinement with $h_{max} = 3$, $c_f = 0.01$ and $r_f = 0.99$. We

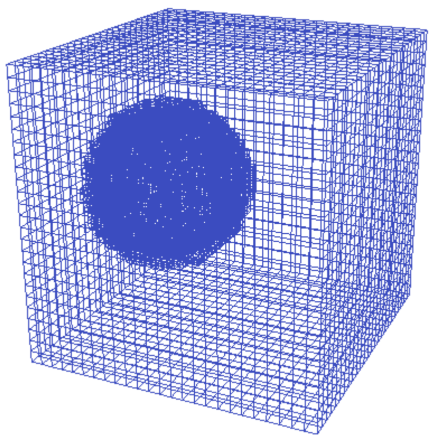


Figure 5. Initial mesh refinement of the advection of a sphere problem.

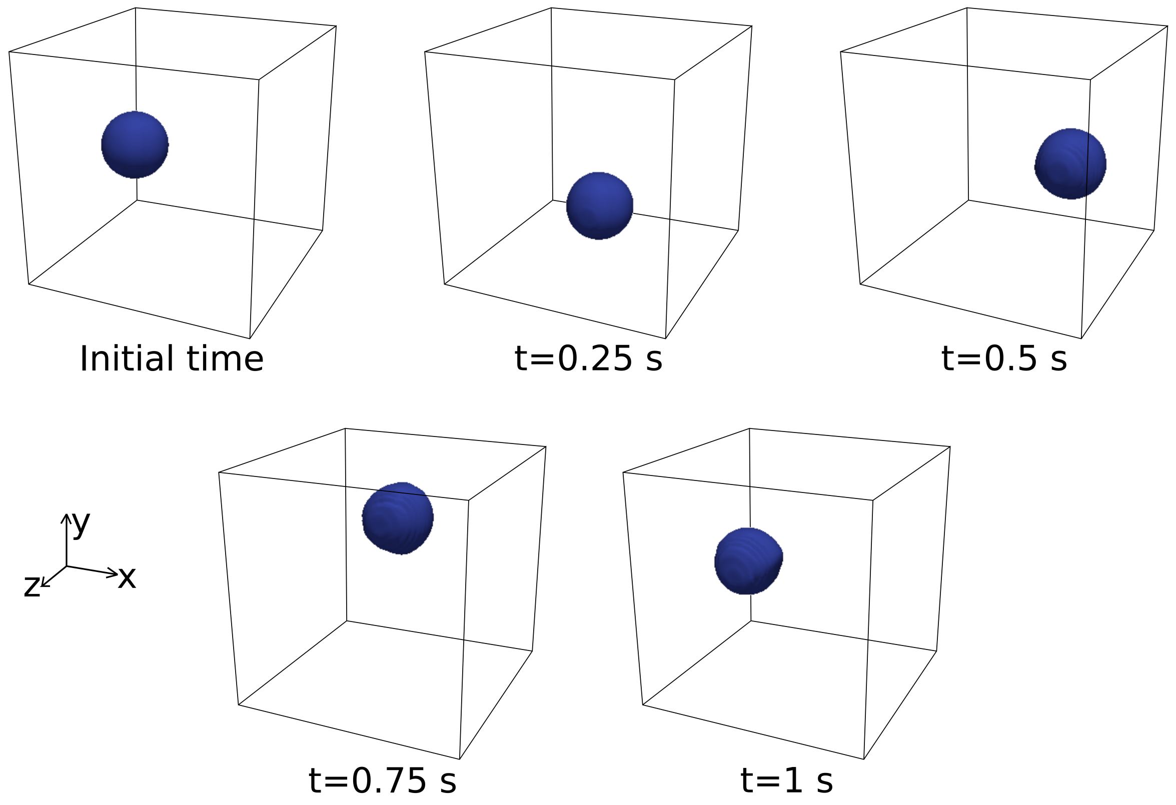


Figure 6. Position of the sphere during one turn.

apply the adaptive mesh refinement every 50 time steps. The initial mesh has 60×60 bilinear quadrilateral elements and after the refinement, the smallest element has 0.0025 m of size. We also initially refine in three levels of the region where the circle is located. We refine more this example because of the sharp corners that loose accuracy with coarses meshes. The time step is 0.0001 seconds. Here we do not use the discontinuity-capturing operator. Figure 8 shows the simulation results at different steps during one revolution of the Zalesak's circle. The zero level-set is highlighted and it is possible to see the mesh adaptivity. In Figure 9 we compare the initial configuration with the final one after one turn. The mass conservation is preserved with a relative error of 0.045%.

We have noted that due the small time-step, the contribution of the stabilization parameters become also too small, and some numerical instabilities were reported. Therefore, we choose a different τ_ϕ for the SUPG contribution, in which we disconsidered the time dependent term. The alternative stabilization

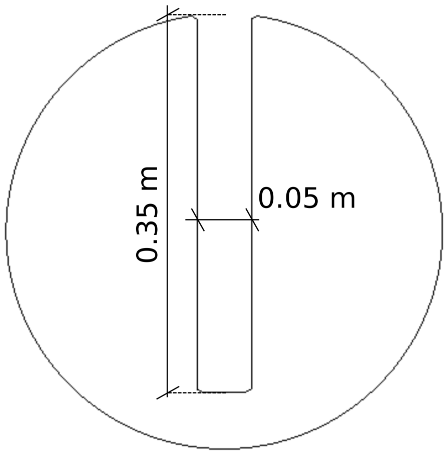


Figure 7. Configuration of the Zalesak's problem.

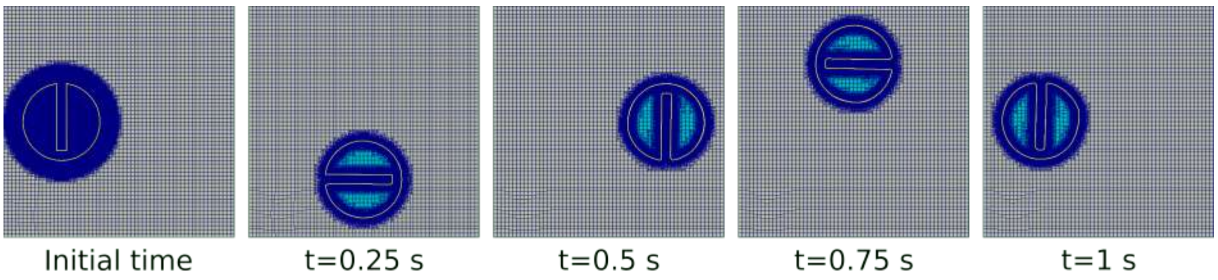


Figure 8. Position of the disk and mesh refinement at different time steps during one turn.

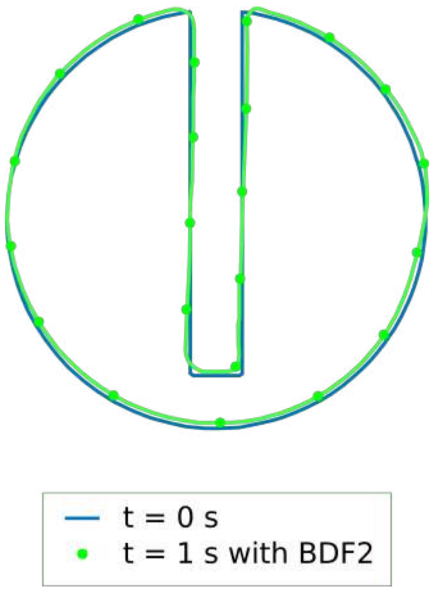


Figure 9. Initial disk configuration (blue line) and final configuration after one turn (green lines with markers)

parameter is given by

$$\tau_\phi = \left(\mathbf{u}^h \cdot \mathbf{G}\mathbf{u}^h\right)^{-\frac{1}{2}} \tag{26}$$

where ϵ_ϕ is the diffusion parameter that we consider equal to zero and $\mathbf{G}$ is a second rank metric tensor

$$\mathbf{G} = \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}}^T \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \tag{27}$$

5.4 Dam-break problem

The level-set is also used to model free surface problems, where the interface between the air and the fluid is the zero isovalue of the level-set function. In this case, we use a heterogeneous Navier-Stokes equations:

$$\begin{aligned} \rho\frac{\partial \mathbf{u}}{\partial t} - \nabla \cdot (2\eta\epsilon(\mathbf{u})) + \nabla p = \rho \mathbf{g} \\ \nabla \cdot \mathbf{u} = 0 \end{aligned} \tag{28}$$

where the density ρ and the viscosity η are defined by

$$\begin{aligned} \rho(\phi) &= \rho_{fluid}\frac{1+H(\phi)}{2} + \rho_{air}\frac{1-H(\phi)}{2} \\ \eta(\phi) &= \eta_{fluid}\frac{1+H(\phi)}{2} + \eta_{air}\frac{1-H(\phi)}{2} \end{aligned} \tag{29}$$

and $H(\phi)$ is the Heaviside function, built as

$$H(\phi) = \begin{cases} 1 \text{ if } \phi > 0 \\ 0.5 \text{ if } \phi = 0 \\ 0 \text{ if } \phi < 0 \end{cases} \tag{30}$$

Here we use the RBVMS formulation for the Navier-Stokes equations as in Grave et al. [25] and Zio et al. [26].

To test the modified level-set function in a two phase problem, we simulate a dam-break. The collapse of a water column is a well-known problem, widely employed to validate free-surface codes based on interface-capturing methods since it has experimental results (see Koshizuka and Oka [27], Martin et al. [28] for details) and several simulation results from different numerical methods (see for instance Elias and Coutinho [13], Löhner et al. [29], Cruchaga et al. [30]). The problem consists of a water column initially sustained by a dam which is suddenly removed. The water falls under the influence of gravity ($g = 9.81m/s^2$), acting vertically, and flows downward until hitting the opposite wall producing a sloshing effect. The model, as shown in Fig. (10), is a box with dimensions $4a \times a \times 2.4a$, where a is a parameter, assumed here to be equal to 0.146 m, following Koshizuka and Oka [27]. The water column has dimensions $a \times a \times 2a$. The density of water is $\rho_w = 1000$ kg/m^3 and the dynamic viscosity $\rho_w = 0.001$ kg/(m s). The density of the air was assumed to be $\eta_a = 1\ kg/m^3$ and the dynamic viscosity $\eta_a = 0.0001$ kg/(m s). The initial structured mesh has 40×24 bilinear quadrilateral elements and after the refinement, the smallest element has 0.00365 m. We use an adaptive mesh refinement based on the level-set function error together with the flow velocities error with $h_{max} = 2$, $c_f = 0.01$ and $r_f = 0.99$. The time step is 0.0001 seconds and the thickness is $E = 0.0292$.

We validate the results plotting the water column leading edge as well as the water column height against the dimensionless time and we compare it with the numerical results from Elias and Coutinho [13] and the experimental ones from Martin et al. [28] as shown in Fig. (11). Moreover, from the snapshots shown in Fig. (12) for the instants 0.2 and 0.4 s, we can qualitatively see the good results obtained with the present approach.

6 Conclusions

We presented the application of the convected level-set method with adaptive mesh refinement in different test cases, as the advection of different phases in two and three dimensions and a free-surface

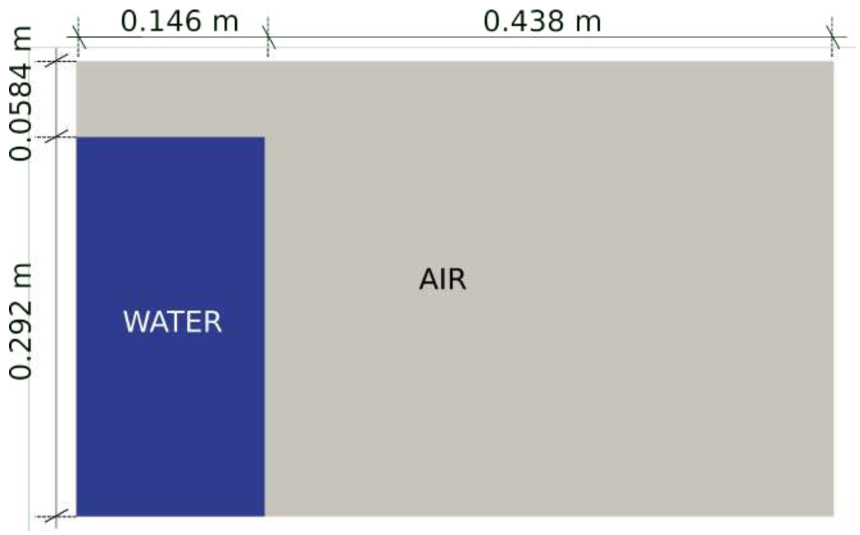


Figure 10. Configuration of the dam-break problem.

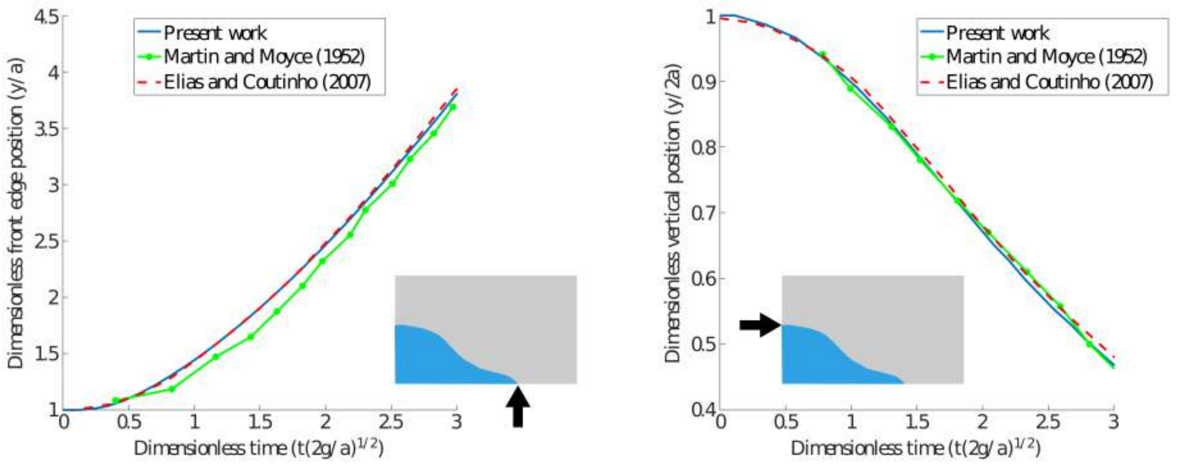


Figure 11. Dam-break validation - (left) leading edge position and (right) water column height.

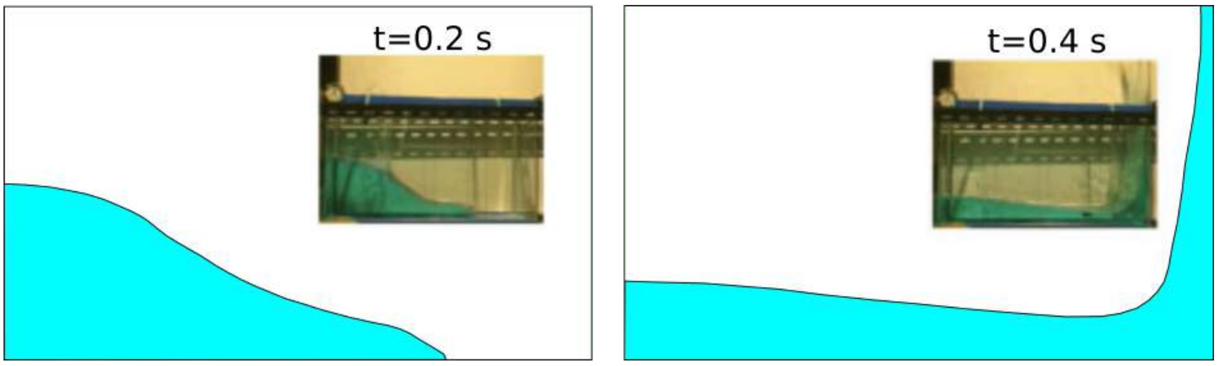


Figure 12. Snapshots for the simulation instant 0.2 and 0.4 s compared with the experimental results presented by Koshizuka and Oka [27]

two-phase fluid problem. The convected level-set method include the reinitialization step in the level-set equation, which avoid an extra step as in the original level-set formulation. We also used a hyperbolic tangential distance function to get a smooth truncation far from the interface.

The results obtained with the convected level-set method had good agreement with the analytical and experimental ones. The mesh adaptivity has shown to be efficient as well. We observed the dependence of the accuracy of the convected level-set method on the mesh size and time discretization method. The mass conservation is also dependent on the discretization of the mesh. As future work, we would like to implement the method with correct kinetic and potential energy behavior proposed by Akkerman and ten Eikelder [31]. They have observed that the results with coarse grids using their method are better than the results obtained only by the convected level-set method, showing the importance of correct energy behavior on two-fluid flow simulations. We also would like to use the convected level-set method to simulate the bed-load transport of sediments in fluid flows.

Acknowledgements

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior- Brasil (CAPES) - Finance Code 001. This work is also partially supported by CNPq, FAPERJ, ANP and Petrobras.

References

[1] Bazilevs, Y., Takizawa, K., & Tezduyar, T. E., 2013. *Computational fluid-structure interaction: methods and applications*. John Wiley & Sons.

[2] Hirt, C. W. & Nichols, B. D., 1981. Volume of fluid (vof) method for the dynamics of free boundaries. *Journal of computational physics*, vol. 39, n. 1, pp. 201–225.

[3] Osher, S. & Sethian, J., 1988. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*, vol. 79, pp. 12–49.

[4] Sussman, M., Almgren, A., Bell, J., Colella, P., Howell, L., & Welcome, M., 1999. An adaptive level set approach for incompressible two-phase flows. *Journal of Computational Physics*, vol. 148, pp. 81–124.

[5] Gibou, F., Fedkiw, R., & R. Caflisch, S. O., 2003. A level set approach for the numerical simulation of dendritic growth. *Journal of Hydraulic Engineering*, vol. 19.

[6] Ville, L., Silva, L., & Coupez, T., 2011. Convected level set method for the numerical simulation of fluid buckling. *International Journal for numerical methods in fluids*, vol. 66, n. 3, pp. 324–344.

[7] Bonito, A. & Guermond, J., 2016. Numerical simulations of bouncing jets. *International Journal for Numerical Methods in Fluids*, vol. 80, n. 1, pp. 53–75.

[8] Dapogny, C., Faure, A., Michailidis, G., Allaire, G., Couvelas, A., & Estevez, R., 2017. Geometric constraints for shape and topology optimization in architectural design. *Computational Mechanics*, vol. 59, n. 6, pp. 933–965.

[9] Yamada, T., Izui, K., Nishiwaki, S., & Takezawa, A., 2010. A topology optimization method based on the level set method incorporating a fictitious interface energy. *Computer Methods in Applied Mechanics and Engineering*, vol. 199, pp. 2876–2891.

[10] Coupez, T., 2006. Reinitialisation convective et locale des fonctions level set pour le mouvement de surfaces et d'interfaces. *Journées Activités Universitaires de Mécanique,*, pp. 140.

[11] Khalloufi, M., Mesri, Y., Valette, R., Massoni, E., & Hachem, E., 2016. High fidelity anisotropic adaptive variational multiscale method for multiphase flows with surface tension. *Computer Methods in Applied Mechanics and Engineering*, vol. 307, pp. 44–67.

[12] Coupez, T., 2007. Convection of local level set function for moving surfaces and interfaces in forming flow. In *AIP Conference Proceedings*, volume 908, pp. 61–66. AIP.

[13] Elias, R. N. & Coutinho, A., 2007. Stabilized edge-based finite element simulation of free-surface flows. *International Journal for Numerical Methods in Fluids*, vol. 54, n. 6-8, pp. 965–993.

[14] Hughes, T. J., 1987. Recent progress in the development and understanding of supg methods with special reference to the compressible euler and navier-stokes equations. *International journal for numerical methods in fluids*, vol. 7, n. 11, pp. 1261–1275.

[15] Bazilevs, Y., Calo, V. M., Tezduyar, T. E., & Hughes, T. J. R., 2007. Yzβ discontinuity capturing for advection-dominated processes with application to arterial drug delivery. *International Journal for Numerical Methods in Fluids*, vol. 54, n. 6-8, pp. 593–608.

[16] Galeão, A. C. & Carmo, E. G. D. D., 1988. A consistent approximate upwind petrov-galerkin method for convection-dominated problems. *Computer Methods in Applied Mechanics and Engineering*, vol. 68, n. 1, pp. 83–95.

[17] Kirk, B. S., Peterson, J. W., Stogner, R. H., & Carey, G. F., 2006. libmesh: a c++ library for parallel adaptive mesh refinement/coarsening simulations. *Journal Engineering with Computers*, vol. 22, n. 3, pp. 237–254.

[18] Balay, S., Abhyankar, S., Adams, M. F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W. D., Karpeyev, D., Kaushik, D., Knepley, M. G., May, D. A., McInnes, L. C., Mills, R. T., Munson, T., Rupp, K., Sanan, P., Smith, B. F., Zampini, S., Zhang, H., & Zhang, H., 2019. PETSc users manual. Technical Report ANL-95/11 - Revision 3.11, Argonne National Laboratory.

[19] Heroux, M. A., Bartlett, R. A., Howle, V. E., Hoekstra, R. J., Hu, J. J., Kolda, T. G., Lehoucq, R. B., Long, K. R., Pawlowski, R. P., Phipps, E. T., et al., 2005. An overview of the trilinos project. *ACM Transactions on Mathematical Software (TOMS)*, vol. 31, n. 3, pp. 397–423.

[20] Ainsworth, M. & Oden, J. T., 2011. *A posteriori error estimation in finite element analysis*, volume 37. John Wiley & Sons.

[21] Peterson, J. W., 2008. *Parallel Adaptive Finite Element Methods for Problems in Natural Convection*. PhD thesis, The University of Texas at Austin.

[22] Rossa, A. L. & Coutinho, A., 2013. Parallel adaptive simulation of gravity currents on the lock-exchange problem. *Computers & Fluids*, vol. 88, pp. 782–794.

[23] Kelly, D. W., De S. R. Gago, J. P., Zienkiewicz, O. C., & Babuska, I., 1983. A posteriori error analysis and adaptive processes in the finite element method: Part i - error analysis. *International Journal for Numerical Methods in Engineering*, vol. 19, n. 11, pp. 1593–1619.

[24] Zalesak, S. T., 1979. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of computational physics*, vol. 31, n. 3, pp. 335–362.

[25] Grave, M., Camata, J. J., & Coutinho, A. L., submitted. Residual-based variational multiscale 2d simulation of sediment transport with morphological changes. *Computers and Fluids*.

[26] Zio, S., da Costa, H. F., Guerra, G. M., Paraizo, P. L., Camata, J. J., Elias, R. N., Coutinho, A. L., & Rochinha, F. A., 2018. Bayesian assessment of uncertainty in viscosity closure models for turbidity currents computations. *Computer Methods in Applied Mechanics and Engineering*, vol. 342, pp. 653–673.

[27] Koshizuka, S. & Oka, Y., 1996. Moving-particle semi-implicit method for fragmentation of incompressible fluid. *Nuclear science and engineering*, vol. 123, n. 3, pp. 421–434.

[28] Martin, J. C., Moyce, W. J., Martin, J., Moyce, W., Penney, W. G., Price, A., & Thornhill, C., 1952. Part iv. an experimental study of the collapse of liquid columns on a rigid horizontal plane.

Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences, vol. 244, n. 882, pp. 312–324.

[29] Löhner, R., Yang, C., & Oñate, E., 2006. On the simulation of flows with violent free surface motion. *Computer Methods in Applied Mechanics and Engineering*, vol. 195, n. 41-43, pp. 5597–5620.

[30] Cruchaga, M. A., Celentano, D. J., & Tezduyar, T. E., 2005. Moving-interface computations with the edge-tracked interface locator technique (etilt). *International Journal for Numerical Methods in Fluids*, vol. 47, n. 6-7, pp. 451–469.

[31] Akkerman, I. & ten Eikelder, M., 2019. Toward free-surface flow simulations with correct energy evolution: an isogeometric level-set approach with monolithic time-integration. *Computers & Fluids*, vol. 181, pp. 77–89.