Journal of Computational Physics 233 (2013) 10–33

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A 3D Unsplit Forward/Backward Volume-of-Fluid Approach and Coupling to the Level Set Method

Vincent Le Chenadec [a,*], Heinz Pitsch [a,b]

[a] Department of Mechanical Engineering, Stanford University, CA 94305, USA
[b] Institute for Combustion Technology, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany

ARTICLE INFO

Article history:
Received 15 November 2011
Received in revised form 21 June 2012
Accepted 4 July 2012
Available online 30 August 2012

Keywords:
Volume-of-Fluid
Level Set
Lagrangian–Eulerian
Interfacial flow
Two-phase flow
Free surface flow
Multiphase flow

ABSTRACT

This paper presents a novel methodology for interface capturing in two-phase flows by combining a Lagrangian–Eulerian Volume-of-Fluid approach with a Level Set method. While the Volume-of-Fluid transport relies on a robust and accurate polyhedral library, any high-order Level Set transport may be used. The method is shown to be less restrictive in terms of CFL conditions than split Volume-of-Fluid methods. Various geometric integration schemes are proposed and tested. For linear velocity fields, mass error is shown to vanish, and to be third order otherwise. The method is validated on 2D and 3D test cases. Conservation properties are shown to be excellent, while geometrical accuracy remains satisfactory even for complex flows such as the primary breakup of a liquid jet.

1. Introduction

Multiphase physics appear in engineering problems ranging from weather forecasting to industrial surface coating. Predicting and analyzing these physics continues to be a challenge in both experimental and computational settings. These challenges are being addressed using improved numerical methods that can lead to a better understanding of the underlying physics, and ultimately result in improved technologies and designs.

A variety of computational models for treating multiphase flows exists, such as Eulerian [1,2] and Lagrangian [3,4] representations, and they involve different levels of modeling. In the Direct Numerical Simulation (DNS) approach [5] that is considered in the presented work, the physics of the flow are all solved for directly. The underlying equations are quite stiff, however, so the development and improvement of numerical methods that treat these flows has been an area of active research over the last two decades [6].

A wide range of methods has emerged, with strengths and weaknesses which may be compared within the scope of a precise application such as the simulation of turbulent multiphase flows. Such flows involve a wide range of scales, and are of particular interest for primary breakup studies [7,8]. Interface capturing methods have proven to be very valuable for these computations. The proposed algorithm, the Hybrid Lagrangian–Eulerian Method for Multiphase flows (HyLEM), therefore lies within this framework, combining a Level Set approach with an unsplit Volume-of-Fluid transport scheme.

HyLEM lies in the scope of Lagrangian–Eulerian methods, and therefore inherits their excellent local conservation properties. By combining both Volume-of-Fluid and Level Set approaches, it also benefits from the excellent curvature accuracy of

* Corresponding author.
E-mail address: vlechena@stanford.edu (V. Le Chenadec).

0021-9991/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2012.07.019

the latter. Furthermore, it extends the use of Lagrangian–Eulerian methods to 3D computations, within a framework that is a priori compatible with unstructured grids. An extensive study of various Volume-of-Fluid transport algorithms within this framework is presented.

This article is structured as follows: a brief description of Level Set and Volume-of-Fluid methods is introduced in Section 2, along with the motivation which leads to the development of the proposed method. A high-level description of HyLEM is therefore presented in Section 3, and the geometric toolbox required in this algorithm is presented in Section 4. The proposed method is implemented in a finite difference incompressible solver, and the details of the implementation are presented in Section 5.

Widespread transport tests and results are provided in Section 6. This section also includes an analytical proof of exact mass conservation for linear velocity fields, along with numerical proof of third order mass error convergence. Finally, some applications are presented in Section 7.

2. Overview and motivation

Rather than explicitly tracking the interface by aligning the mesh with it, such as pure Lagrangian methods [9], interface capturing techniques rely on an implicit representation of the interface. The two most widespread interface capturing methods, Level Set and Volume-of-Fluid, differ in how the interface is represented. This choice results in various strengths and weaknesses, which are briefly described below in the context of turbulent flow computations.

2.1. Level Set methods

Level Set methods rely on the representation of the liquid/gas interface by an iso-surface of a smooth function, usually defined as the signed distance function,

$$G(\mathbf{x},t)=\begin{cases}-|\mathbf{x}-\mathbf{x}_\Gamma| & \text{if } \mathbf{x} \text{ is in the gas side at time } t,\\ +|\mathbf{x}-\mathbf{x}_\Gamma| & \text{if } \mathbf{x} \text{ is in the liquid side at time } t,\end{cases} \tag{1}$$

where $\mathbf{x}_\Gamma$ is the closest point on the interface to $\mathbf{x}$.

Their simplicity makes them suitable for very accurate transport schemes. Their performance, however, is usually undermined by their poor mass conservation properties.

2.1.1. Governing equations

In the absence of mass transfer, the evolution of the Level Set function G is governed by,

$$\frac{\mathrm{D}G}{\mathrm{D}t}=\frac{\partial G}{\partial t}+\mathbf{u}\cdot\boldsymbol{\nabla}G=0, \tag{2}$$

where $\mathbf{u}$ is the fluid velocity.

Long time integration leads in general to the deviation of the G-field from a signed distance. For flows with capillary effects, this can be detrimental to the accuracy of the curvature computation,

$$\kappa=-\boldsymbol{\nabla}\cdot\left(\frac{\boldsymbol{\nabla}G}{|\boldsymbol{\nabla}G|}\right). \tag{3}$$

An additional step, known as the re-initialization, is therefore usually used, in which one ideally modifies the G-field without affecting the location of the $G=0$ surface. This is done by solving the Eikonal equation $|\boldsymbol{\nabla}G|=1$, which can be solved iteratively [10] or geometrically [11].

2.1.2. Advantages and shortcomings

The smoothness of the G field results in the excellent geometric accuracy of Level Set methods: no explicit reconstruction of the interface is required, and topology changes are handled automatically. In addition, time integration may be performed using arbitrary high-order transport schemes, or spectral methods such as the Discontinuous Galerkin method [12]. The hyperbolic nature of Eq. (2) has also been exploited in particle Level-Set methods [13] and semi-Lagrangian methods [14].

However, the simplicity of Level Set implementations comes at a cost, namely that there is no inherent mass conservation in the Level Set equation. In fact, Level Set methods are known to lose or gain liquid mass in positive and negative curvature regions respectively. This is illustrated in Fig. 1, where an initially sinusoidal interface is transported with unit velocity along a periodic direction. In this simple example, mass is nonphysically transferred from positive to negative curvature regions, while the global mass error stays very small (less than 0.01% for this particular case where a fifth order WENO interpolation was used). This illustrates the potential errors in mass conservation Level Set methods are subject to.

2.2. Volume-of-Fluid methods

Volume-of-Fluid methods on the other hand aim at transporting the indicator function

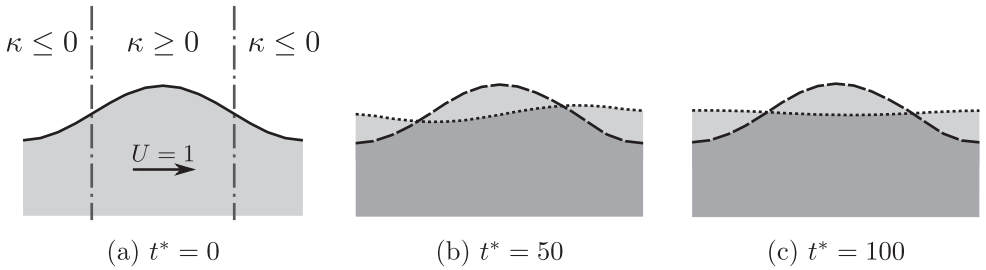


Fig. 1. Spurious mass transfer (*dashed line*: exact solution, *dotted line*: Level Set).

$$f(\mathbf{x}, t) = \begin{cases} 0 & \text{if } \mathbf{x} \text{ is in the gas side at time } t, \\ 1 & \text{if } \mathbf{x} \text{ is in the liquid side at time } t. \end{cases} \tag{4}$$

Discretely, the information effectively stored is a normalized estimate of the volume-averaged indicator function, namely the volume fraction

$$F(\Omega, t) = \frac{\int_\Omega f(\mathbf{x}, t)\, d\mathbf{x}}{\int_\Omega d\mathbf{x}}. \tag{5}$$

For the volume fraction field to be meaningful, a sharp profile must be maintained at all times. As described below, satisfying this condition is precisely the hurdle that Volume-of-Fluid methods face.

2.2.1. Methodology

The potential of Volume-of-Fluid methods to conserve mass is obvious from the choice of variables. Discretely however, it is not straightforward to ensure conservation. The most widespread Volume-of-Fluid schemes rely on a purely Eulerian discretization of

$$\frac{\partial f}{\partial t} + \nabla \cdot (f\mathbf{u}) = f \nabla \cdot \mathbf{u}, \tag{6}$$

where the right-hand side is zero for solenoidal velocity fields.

Discretely, this is done by estimating the fluxes at each element face $\partial\Omega$,

$$\Phi = \int_{t^n}^{t^{n+1}} \int_{\partial\Omega} (f\mathbf{u}) \cdot \mathbf{n}\, dA\, dt. \tag{7}$$

The knowledge of the volume fraction F within each element, however, is not sufficient to accurately estimate Φ. In order to do so, one needs to estimate the liquid fluid distribution within each element: this is known as the reconstruction step.

Given an estimate of the indicator function at the cell level, estimating Φ still requires complex geometrical operations. In order to simplify this step, a common approach is to use dimensional splitting, which reduces Eq. (6) to a sequence of one-dimensional transport steps; such an approach is referred to as Split Volume-of-Fluid [15]. Geometrically more complex, Unsplit Volume-of-Fluid algorithms have also been developed [16].

An alternative to pure Eulerian methods is to solve the weak form of Eq. (6) over an Eulerian mesh. For a solenoidal velocity field, Eq. (6) can be written

$$\frac{\mathrm{D}_m}{\mathrm{D}_m t} \int_{\Omega_m(t)} f\, d\mathbf{x}(t) = 0, \tag{8}$$

where the subscript m is used to denote material control volumes and material derivatives. The complexity of such an approach, referred to as Lagrangian–Eulerian Volume-of-Fluid, is similar to the Unsplit Volume-of-Fluid method, and the discrete equivalence of both formulations can be shown.

2.2.2. Advantages and shortcomings

The main and most obvious advantage of Volume-of-Fluid methods is the excellent mass conservation property inherent to the definition of the transported variable. A conservative discretization which maintains boundedness of the F-field ensures exact mass conservation. Provided that it is geometrically accurate, such a scheme seems to fulfill all requirements to a robust and accurate interface capturing scheme (it will be shown later that this notion of geometrical accuracy is precisely what deteriorates the properties of Volume-of-Fluid methods). As far as the flow solver is concerned, Volume-of-Fluid implementations can also lead naturally to momentum conservation, known to be suitable for flows involving large density ratios [17].

Volume-of-Fluid methods unfortunately also suffer from shortcomings. The added complexity with respect to the Level Set methods is often a practical impediment. While the complexity can be dramatically reduced using dimensional splitting, the accuracy of such implementations is found to be deteriorated by the flotsam/jetsam phenomenon, known to lead to spurious sub-cell structures. These structures still occur in unsplit formulations, but to a lesser extent. In addition, mass conservative formulations are not straightforward to derive. If simple conservative directional splitting algorithms exist [18], their strenuous time-step restriction and directional splitting assumption hinder their applicability to efficient computation in complex geometries. A final hurdle is related to the low order reconstruction step (usually linear), which is known to deteriorate the accuracy of the curvature computation. It has to be noted that while a parabolic reconstruction method has been previously developed [19], its extension to unsplit transport is extremely challenging.

2.3. Improvement strategy

The brief introduction presented above highlighted various properties addressing the two main challenges involved in turbulent two-phase flow simulations, conservation and geometric accuracy.

In the inviscid limit of the Navier–Stokes equations, mass and momentum are indeed conserved (this is referred to as primary conservation), along with higher order moments such as kinetic energy (this is known as secondary conservation). It has been shown that discretely satisfying these conservation properties leads to robust and accurate numerical methods [20]. Mass and momentum conservative schemes are therefore essential for accurate turbulent flow computations, which suggests that maintaining a sharp and mass conservative representation of the interface is a requirement for robust multiphase flow computations.

The second challenge, geometrical accuracy, is strongly coupled to the range of scales involved in turbulent multiphase flows. Of particular importance are the scales resulting from the interface dynamics and the capillary effects, which are an additional constraint to consider. Capillary forces are related to the interface topology. Accurate curvature computation is therefore crucial, especially in regions of the flow where the interface is under-resolved since the capillary force is inversely proportional to the interface curvature radius. This unfortunate incompatibility suggests unstructured grids and support for local refinement as two essential strategies to accurately resolve as wide of a spectrum as a computational domain can. A suitable interface capturing technique should therefore be compatible with both of these strategies.

Over the years, developments have led to achieving some of these goals. Spectrally refined Level Set [12,14], Particle Level Set [13] and Conservative Level Set methods [21,22] led to improvement of the conservation properties of Level Set methods, while techniques such as the osculating circle [23], PROST [19] and height function [24–29] aimed at improving the curvature computation accuracy in Volume-of-Fluid methods.

An interesting approach introduced by Sussman and Puckett [30] consists in combining Level Set and Split Volume-of-Fluid methods. The idea is to benefit first from the conservation properties of Volume-of-Fluid methods, and second from the geometrical accuracy of Level Set methods to accurately transport the volume fraction field and compute the curvature. The original split algorithm was developed for structured grids, but later extended to two-dimensional triangular meshes [31] within a Lagrangian–Eulerian framework.

The proposed method extends the Coupled Level Set and Volume-of-Fluid method to an unsplit transport framework in three-dimensions for arbitrary elements. It is a priori compatible with unstructured grids and local refinement. This has not been implemented, but will instead be the focus of future work. An extensive and comprehensive characterization of various Lagrangian–Eulerian transport schemes is presented. The resulting algorithms are shown to conserve mass exactly for linear velocity fields (in both space and time), to maintain linearity, and to be second order geometrically accurate.

3. Hybrid Lagrangian–Eulerian method for multiphase flows

The main hurdle in computing free surface flows stems from the fact that the thickness of the interface between the two media is small compared to any obtainable mesh resolution. The accurate numerical representation of that interface is therefore challenging. In front-tracking methods [9], this is done very intuitively: the mesh is aligned with the interface, and the complexity remains reasonable in the case of simple geometries. It becomes tedious, however, when merging or breaking of structures occurs, and the quality of the mesh can rapidly deteriorate over time. On the other hand, front-capturing methods [15], some of which were described in Section 2, aim at advancing conservation laws by estimating fluxes over a target mesh.

Lagrangian–Eulerian methods [32,33,31] combine both descriptions to improve either one alone. The idea is to start a time-step with a purely Lagrangian method by using the flow velocity to deform an initial mesh over which the volume fraction field is known, and to project the resulting information onto a target mesh. This allows for the mesh to stay well conditioned at any time. This projection, also known as remapping or rezoning, can be done at every step or for selected iterations.

This simple idea offers some flexibility in terms of time integration, but also represents some challenges as far as its implementation is concerned. In the HyLEM method proposed here, the computational mesh is first deformed in time in order to advance the volume fraction field. The remapping to the original mesh is assisted by the solution of a Level Set equation, which is solved independently on the original target mesh. This section provides a high-level description of the new HyLEM method. Two time-integration schemes are described for the transport of the volume fraction field using the Level

Set function. The two-way coupling between the volume fraction and Level Set field is achieved by a Level Set correction procedure, also introduced in this section.

Section 4 provides detailed description of the routines necessary to perform the required operations.

3.1. Level Set

As pointed out in Section 2.1, the Level Set function G is advanced from time n to time $n+1$ on the original target mesh by discretely solving

$$\frac{\partial G}{\partial t} + \nabla \cdot (G\mathbf{u}) = 0. \tag{9}$$

The method used to solve Eq. (9) does not affect the HyLEM algorithm, and any desired scheme can be used. A reinitialization step is still necessary, however, and has to be performed at selected times using any method leading to the solution of the Eikonal equation $|\nabla G| = 1$.

The solution of the Level Set field will be used in the following projection steps and the consistency with the volume fraction field is described below.

3.2. Forward projection

A first integration scheme, referred to as Forward projection, follows from a simple mass balance. Consider a material control volume $\Omega_{\mathcal{E}}^n$ corresponding to an element of the target mesh at time n ($\mathcal{E}^n$). When tracked up to time $n+1$ with the flow velocity, this control volume results in $\Omega_{\mathcal{L}}^{n+1}$. The union of all these elements forms the Lagrangian mesh at time $n+1$, $\mathcal{L}^{n+1}$. This is shown in Fig. 2.

Because this volume $\Omega_{\mathcal{L}}^{n+1}$ is assumed to be a "material" control volume, the mass it contains is equal to the mass contained at time n in $\Omega_{\mathcal{E}}^n$. In addition, because the densities of both liquid and gas are constant, the volume fractions are also the same. This was expressed in Eq. (8), which can also be rewritten

$$F_{\mathcal{L}}^{n+1} \equiv \frac{\int_{\Omega_{\mathcal{L}}^{n+1}} f(\mathbf{x}, t^{n+1})\, dV}{\int_{\Omega_{\mathcal{L}}^{n+1}} dV} = \frac{\int_{\Omega_{\mathcal{E}}^{n}} f(\mathbf{x}, t^{n})\, dV}{\int_{\Omega_{\mathcal{E}}^{n}} dV} \equiv F_{\mathcal{E}}^{n}. \tag{10}$$

The values $F_{\mathcal{L}}^{n+1}$ are, however, not defined on the same mesh as $F_{\mathcal{E}}^{n+1}$. Because the choice is made in the proposed algorithm to remap at every iteration, the volume fraction field must always be interpolated back onto the target mesh $\mathcal{E}^{n+1}$ in an accurate and conservative fashion. This is done by computing for each element $\Omega_{\mathcal{E}}^{n+1}$

$$F_{\mathcal{E}}^{n+1} = \left(\int_{\Omega_{\mathcal{E}}^{n+1}} dV \right)^{-1} \sum_{\Omega_{\mathcal{L}}^{n+1}} \left(\int_{\Omega_{\mathcal{L}}^{n+1} \cap \Omega_{\mathcal{E}}^{n+1}} f(\mathbf{x}, t^{n+1})\, dV \right), \tag{11}$$

which denotes how the volume fraction on a target mesh element $\Omega_{\mathcal{E}}^{n+1}$ is computed by looping over all elements of the Lagrangian mesh $\Omega_{\mathcal{L}}^{n+1}$ and integrating the indicator function over the intersecting volume.

The discretization of Eqs. (10) and (11) requires three components, each of which is described in the following:

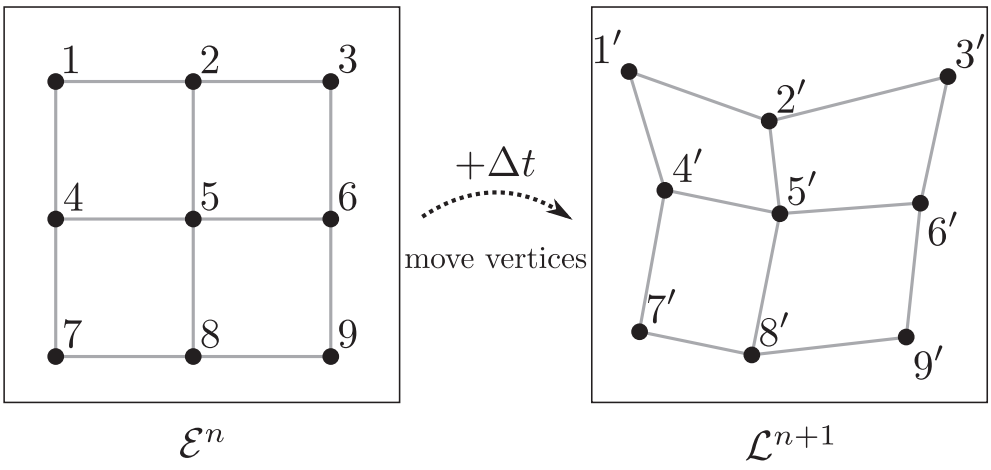


Fig. 2. Forward projection: vertex advection.

1. Definition of an estimate of each Lagrangian element $\Omega_{\mathcal{L}}^{n+1}$ (mesh deformation).
2. Approximation of the indicator function $f(\mathbf{x}, t^{n+1})$ at time $n+1$ over each Lagrangian element $\Omega_{\mathcal{L}}^{n+1}$ (interface reconstruction).
3. Computation of $F_{\mathcal{E}}^{n+1}$ from Eq. (11) using the estimate $f(\mathbf{x}, t^{n+1})$ over each Lagrangian element $\Omega_{\mathcal{L}}^{n+1}$ (remapping).

3.2.1. Mesh deformation

In order to discretely transport each material control volume $\mathcal{E}^n$, each target vertex located at $\mathbf{x}_{\mathcal{E}}^n$ is transported to t^{n+1} by discretely solving,

$$\begin{cases} \mathbf{x}(t^n) = \mathbf{x}_{\mathcal{E}}^n, \\ \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{u}(\mathbf{x}, t). \end{cases} \tag{12}$$

For reasons detailed in Section 6, the conservation of the Forward projection is strongly related to truncation error in the discretization of Eq. (12).

As shown in Fig. 2, the resulting set of vertices defines the new Lagrangian mesh $\mathcal{L}^{n+1}$. From Eq. (10), the volume fraction in each element $\Omega_{\mathcal{L}}^{n+1}$ is known, and needs to be interpolated back onto the target mesh $\mathcal{E}^{n+1}$. Prior to this, a local reconstruction of the liquid spatial distribution has to be computed in order to maintain a sharp interface.

3.2.2. Interface reconstruction

A piecewise linear reconstruction of the interface is performed for all interfacial Lagrangian cells (those satisfying $0 < F_{\mathcal{L}}^{n+1} < 1$). The procedure is as follows: first, the interface normal along with an estimate of the distance to the interface are computed from the neighboring values of the Level Set field G^{n+1}. This step is carried using the least square minimization described in Section 4. Second, the resulting plane is translated by modifying the distance value until the volume fraction on the liquid side of the plane matches $F_{\mathcal{L}}^{n+1}$. The procedure is illustrated in Fig. 3 for a single element $\Omega_{\mathcal{L}}^{n+1}$. Details are also provided in Section 4.

3.2.3. Remapping

The last step of the Forward projection requires to discretely solve Eq. (11). From this equation, it is obvious that only liquid patches overlapping with a given element $\Omega_{\mathcal{E}}^{n+1}$ will affect the corresponding volume fraction.

After initializing $F_{\mathcal{E}}^{n+1}$ to zero, the remapping step therefore consists in looping over all Lagrangian elements satisfying $F_{\mathcal{L}}^{n+1} > 0$ and computing the volume of the intersection of the enclosed liquid patch with all neighboring $\Omega_{\mathcal{E}}^{n+1}$, as illustrated in Fig. 4.

The final step simply consists in normalizing the resulting value in each Eulerian cell by its volume, eventually leading to $F_{\mathcal{E}}^{n+1}$.

3.3. Backward projection

An alternative way of transporting the volume fraction field results from following the material trajectories backward in time. One therefore needs to first consider a material control volume at time $n+1$, such as an element of the target mesh at time $n+1$ ($\Omega_{\mathcal{E}}^{n+1} \in \mathcal{E}^{n+1}$). Projecting this element backward in time with the fluid velocity results in a new element, denoted $\Omega_{\mathcal{L}}^{n}$. The union of all these elements forms the Lagrangian mesh at time n, $\mathcal{L}^n$.

As in the Forward projection, since only material trajectories are used to track the elements backward in time, mass is conserved for each of these elements. In addition, because the density in each phase is constant and no mass transfer occurs, the volume fraction is conserved

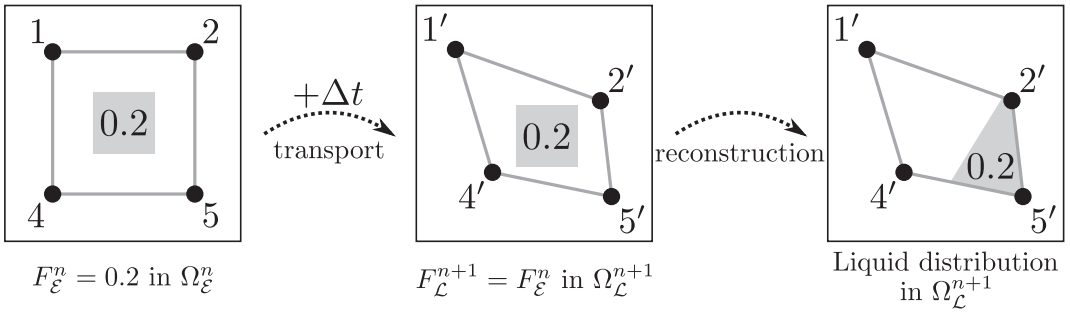


Fig. 3. Local liquid patches reconstruction.

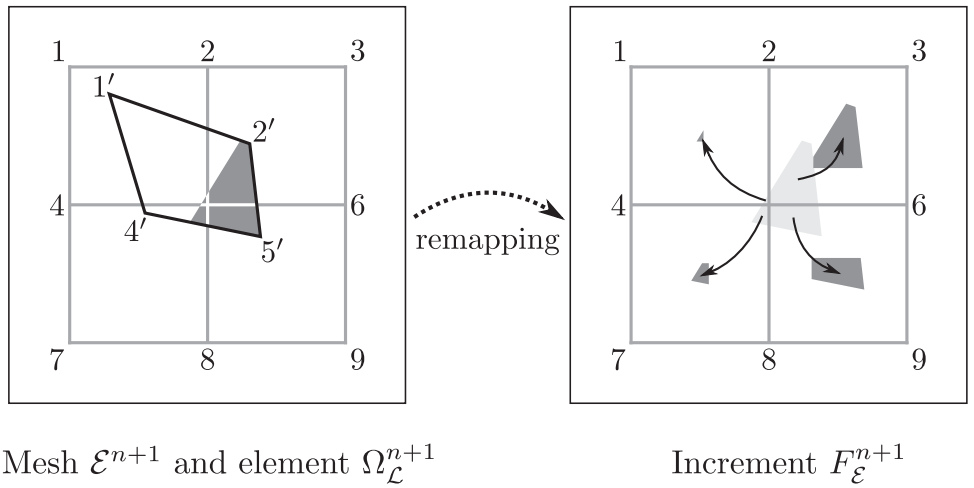


Fig. 4. Remapping and volume fraction field update in the Forward projection scheme.

$$F_{\mathcal{E}}^{n+1} \equiv \frac{\int_{\Omega_{\mathcal{E}}^{n+1}} f(\mathbf{x}, t^{n+1})\, dV}{\int_{\Omega_{\mathcal{E}}^{n+1}} dV} = \frac{\int_{\Omega_{\mathcal{L}}^{n}} f(\mathbf{x}, t^{n})\, dV}{\int_{\Omega_{\mathcal{L}}^{n}} dV} \equiv F_{\mathcal{L}}^{n}. \tag{13}$$

In order to advance the volume fraction field $F_{\mathcal{E}}^{n+1}$, Eq. (13) shows that one simply needs to compute $F_{\mathcal{L}}^{n}$, which can be obtained from

$$F_{\mathcal{L}}^{n} = \left(\int_{\Omega_{\mathcal{L}}^{n}} dV \right)^{-1} \sum_{\Omega_{\mathcal{E}}^{n}} \int_{\Omega_{\mathcal{L}}^{n} \cap \Omega_{\mathcal{E}}^{n}} f(\mathbf{x}, t^{n})\, dV. \tag{14}$$

Discretely solving Eq. (14) requires similar steps to those performed for the Forward projection, but their sequence is organized slightly differently:

1. Approximation of the indicator function $f(\mathbf{x}, t^n)$ at time n over each element $\Omega_{\mathcal{E}}^{n}$ (interface reconstruction).
2. Definition of an estimate of each Lagrangian element $\Omega_{\mathcal{L}}^{n}$ (mesh deformation).
3. Solution of Eq. (14) using the approximation $f(\mathbf{x}, t^n)$ and all Lagrangian elements $\Omega_{\mathcal{L}}^{n}$ (remapping).

These three steps are described and illustrated in more details below.

3.3.1. Interface reconstruction

One of the two pre-requisites to solving Eq. (14) is the knowledge of an estimate of the local liquid distribution within each element $\Omega_{\mathcal{E}}^{n}$ satisfying $0 < F_{\mathcal{E}}^{n} < 1$. Assuming the piecewise linearity of the interface, this is done for each of these ele-

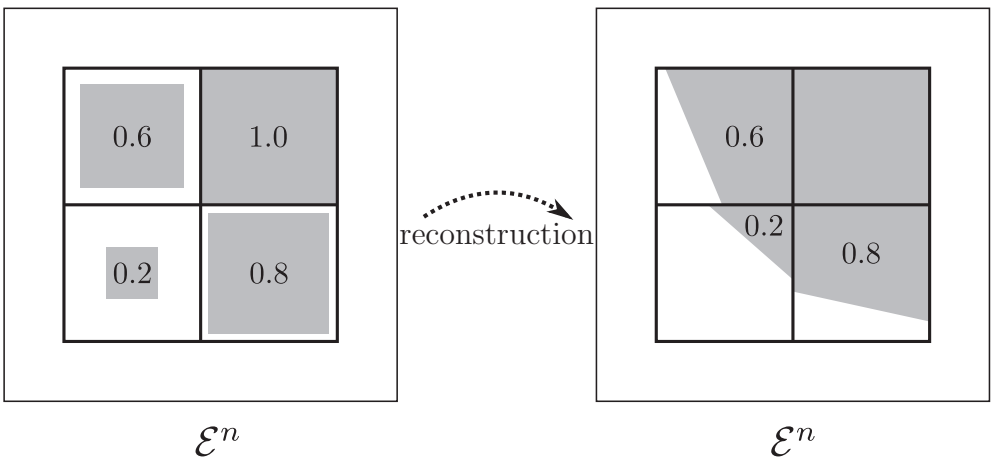


Fig. 5. Reconstruction step required for backward projection step.

ments by estimating the interface normal from the Level Set field G^n as described in Section 3.2.2 and adjusting the normal plane to match $F^n_{\mathcal{E}}$.

The difference with the reconstruction used for the Forward projection is that this is now done over the Eulerian mesh at time n, while it was previously done for the Lagrangian mesh at time $n+1$. The resulting information is illustrated in Fig. 5, and will to be used to discretely solve Eq. (14) once the Lagrangian mesh at time n is defined.

3.3.2. Mesh deformation

Each element $\Omega^n_{\mathcal{L}}$ is defined by transporting the vertices adjacent to the corresponding element at time $n+1$ backward in time. Each target vertex located at $\mathbf{x}^{n+1}_{\mathcal{E}}$ is therefore transported to t^{n+1} by discretely solving

$$\begin{cases} \mathbf{x}(t^{n+1}) = \mathbf{x}^{n+1}_{\mathcal{E}}, \\ \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{u}(\mathbf{x}, t) \end{cases} \tag{15}$$

from time $n+1$ to time n in the same fashion as Eq. (12).

As shown in Fig. 6, this defines the Lagrangian mesh at time n, $\mathcal{L}^n$.

3.3.3. Remapping

The last component of the Backward projection aims at estimating $F^n_{\mathcal{L}}$ by discretely solving Eq. (14). Geometrically, it simply consists in computing the volume of liquid embedded in $\Omega^n_{\mathcal{L}}$, and to scale it with its volume. This is shown in Fig. 7.

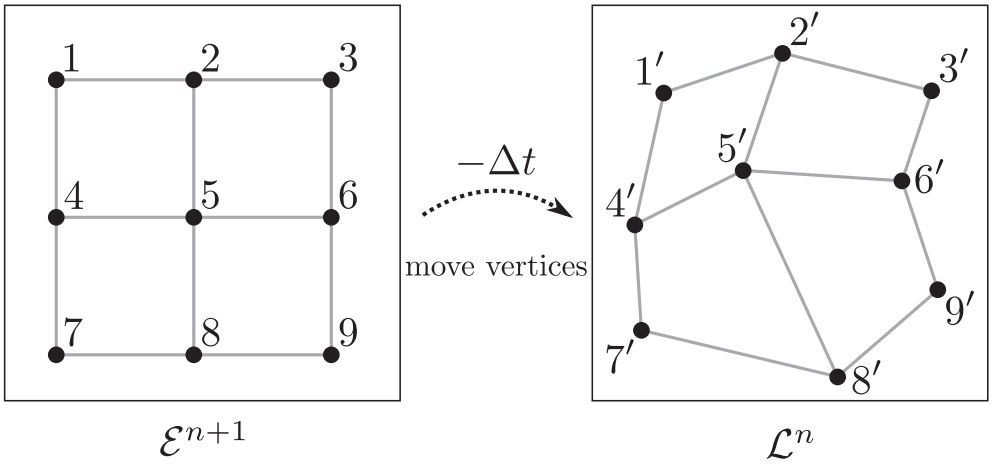


Fig. 6. Backward projection: vertex transport.

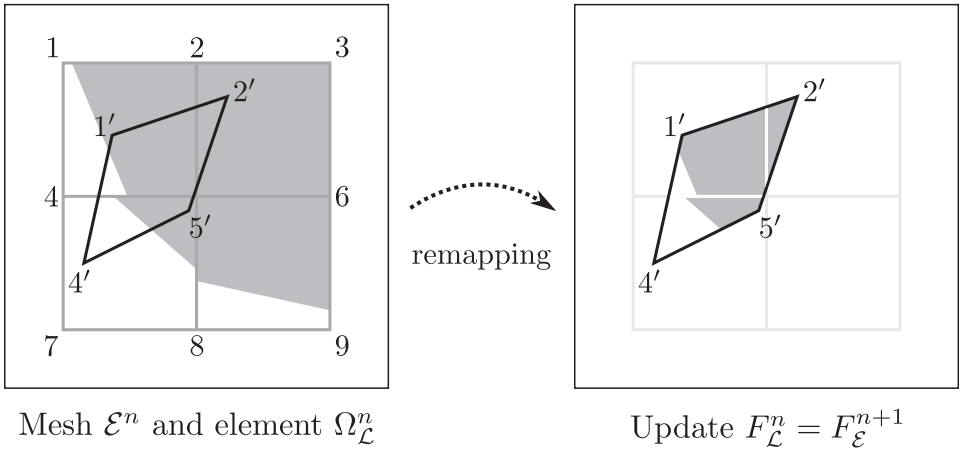


Fig. 7. Backward projection: remapping.

3.4. Level Set correction

The coupling described in the Forward and Backward projections is only one-way: the Level Set field at time n was used to estimate the indicator function $f(\mathbf{x}, t^n)$, and at time $n+1$ to estimate $f(\mathbf{x}, t^{n+1})$. This, however, is not sufficient to accurately solve the flow. The reason is that the accumulation of errors due to the mass loss inherent to Level Set methods increases the discrepancies between G and F. The role of the volume fraction is to provide an anchor to the Level Set field.

One therefore needs to somehow correct the Level Set field close to the interface. The interface reconstruction over the Eulerian mesh provides the information to do that. In the original Coupled Level Set and Volume-of-Fluid method [30], the Level Set field was reset to the exact (computed from the reconstruction) signed distance function at selected times and according to specific rule. This was found in the present study to introduce spurious oscillations, which were reduced by weakening the F to G coupling using the following relaxation formula,

$$G \leftarrow \omega G + (1-\omega)d, \tag{16}$$

where d was taken as the exact signed distance function.

The weight ω is important for the following two reasons. First, even for well-resolved regions (low curvature κ), the discrepancy $|G-d|$ is not expected to be zero, but to scale as,

$$|G-d| \propto \kappa. \tag{17}$$

However, in capillary dominant flows, even for low curvature regions, too tight of a coupling leads to an oscillatory behavior resulting from the mismatch between the F and G representations. This suggests the introduction a discrepancy tolerance h' below which (when $|G-d| < h'$) the Level Set field G should not be corrected.

A variety of weighting formulas were derived and tested, with a particular focus on their effects on unresolved structures, and the following simple formula was found to lead to very reasonable results,

$$\omega = \exp\left[-\alpha \max\left(\frac{|G-d|}{h'} - 1, 0\right)^2\right], \tag{18}$$

where h' was taken as 1% of the mesh size h for example. The square power aims at smoothing the relaxation, and α is a simple constant determining how fast the relaxation is made (a value of 10 was used).

4. Polyhedral library

This section presents various routines used to perform polyhedral operations. Most of the ideas are taken from the field of Computational Geometry [34].

4.1. Lagrangian element definition

In Section 3, a brief description of how the Lagrangian meshes are built was provided: the vertices of the target mesh are transported backward or forward in time, and the connectivity is maintained. In other words, if two vertices were linked by an edge, their Lagrangian images still are. The definition of the Lagrangian mesh follows easily.

This definition, however, is not sufficient to properly define the Lagrangian meshes. But this can be addressed by means of a triangulation of each element.

4.1.1. Challenges

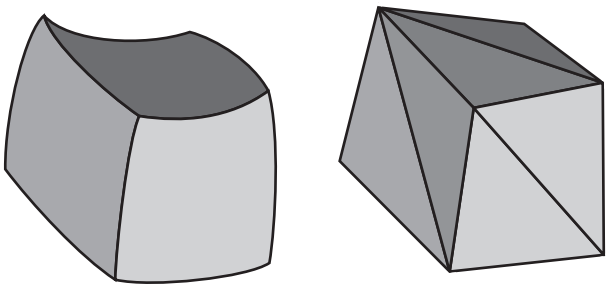
The challenges to be addressed appear in different ways throughout the proposed algorithm and are related to

1. *Topology*: The vertex transport does not include any constraint that keeps Lagrangian edges from crossing. This can lead to element folding.
2. *Polyhedra algebra*: The routines used for polyhedra intersection and volume computation described below all operate on convex polyhedra. Accordingly, concave elements are not a priori supported.
3. *Mesh well-posedness*: The vertex transport does not maintain co-planarity. This is problematic in 3D, as illustrated in Fig. 8(a) and (b). Consider, for example, the vertices adjacent to a quadrilateral face of the target mesh. After projection, the plane defined by any triplet of vertices does not necessarily contain the fourth vertex: the projected face is not planar. This needs to be addressed, since the Lagrangian mesh should not suffer from gaps or overlaps.

All these challenges will be addressed with a proper element triangulation described next.

4.1.2. Element triangulation

An easy fix to get around the folding and concavity hurdles is to triangulate each mesh element. In order to reduce the cost of the remapping step, the number of simplices used to triangulate each element should be minimal. One can use each



(a) Exact deformation. (b) Triangulation.

Fig. 8. Face deformation for a hexahedral element.

simplex throughout the intersection and volume computation routines since they always remain convex, but the resulting volume has to be subtracted if the simplex flipped.

In D dimensions, given an Eulerian simplex bounded by the vertices $(\mathbf{x}_{\mathcal{E}0}, \ldots, \mathbf{x}_{\mathcal{E}D})$ and the corresponding Lagrangian simplex bounded by $(\mathbf{x}_{\mathcal{L}0}, \ldots, \mathbf{x}_{\mathcal{L}D})$, the vertex transport leads to flipping if the signature ς is negative

$$\varsigma = \mathrm{Vol}(\mathbf{x}_{\mathcal{E}0}, \ldots, \mathbf{x}_{\mathcal{E}D}) \cdot \mathrm{Vol}(\mathbf{x}_{\mathcal{L}0}, \ldots, \mathbf{x}_{\mathcal{L}D}) \leqslant 0, \tag{19}$$

where the formula of a simplex in D dimensions was used

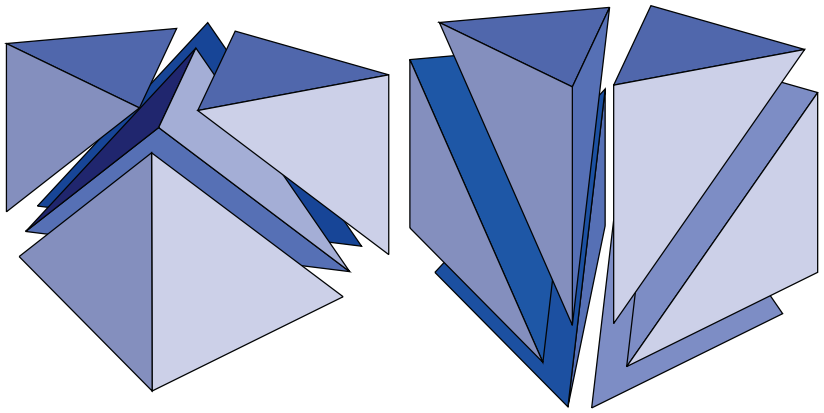
$$\mathrm{Vol}(\mathbf{x}_0, \ldots, \mathbf{x}_D) = \frac{\det(\mathbf{x}_1 - \mathbf{x}_0, \ldots, \mathbf{x}_D - \mathbf{x}_0)}{D!}. \tag{20}$$

As shown in Fig. 8(a), the projection of a simple hexahedral Eulerian element results in an object which is not a polyhedron anymore, since its faces are not planar. The framework presented so far cannot cope with this type of object, which again motivates the need for triangulation in general. The triangulation of the Eulerian element and the application of the same connectivity to the projected vertices leads to a set of simplices similar to the one illustrated in Fig. 8(b). The volume triangulation also results in a face triangulation which, for the mesh to be well-posed, has to be identical for both of the elements on either side of it.

A minimum of five tetrahedra is required to triangulate a hexahedron as shown in Fig. 9(a). The resulting face triangulation is not cell-to-cell compatible however: considering two opposing faces in Fig. 9(a), the diagonals along which each face is decomposed are not aligned. This decomposition for two neighboring cells will result in the Lagrangian cells to either overlap or be non-adjacent, ultimately leading to mass errors.

A decomposition such as the one shown in Fig. 9(b), however, does not suffer from this limitation, making the algorithm applicable to hexahedral meshes.

The proposed method therefore natively supports tetrahedral elements, but is easily applicable to more general elements and to sub-cell resolution provided a cell-to-cell compatible triangulation is made available.



(a) Using 5 tetrahedra. (b) Using 6 tetrahedra.

Fig. 9. Two different triangulations of a hexahedron.

4.2. Interface reconstruction

As stated previously, a piecewise discontinuous representation of the interface is used, which presents some drawbacks with respect to higher order reconstruction methods such as PROST [19]. However, given a quadratic fit to the interface, PROST requires its integration over a right hexahedron. For a Lagrangian–Eulerian scheme, this is required over complex polyhedra, which is very tedious to implement. Instead, the approach adopted here is to restrict the accuracy to linear reconstruction, while focusing on the ability to add sub-cell resolution in order to reduce the error related to the low order interface reconstruction.

In the backward (resp. forward) projection, interface reconstruction is required for any Eulerian element satisfying $0 < F_{\mathcal{E}}^{n} < 1$ (resp. Lagrangian element satisfying $0 < F_{\mathcal{L}}^{n+1} < 1$). Accordingly, for these elements, the objective is to fit a plane,

$$\mathcal{P} = \{\mathbf{n} \cdot (\mathbf{x} - \mathbf{x}_c) + d = 0\}, \tag{21}$$

to the $G = 0$ iso-surface ($\mathbf{x}_c$ is the origin of the coordinate system).

4.2.1. Normal computation and intercept initial guess

The first step in the procedure is to estimate the interface normal $\mathbf{n}$ and the distance to the interface d by minimizing the following error,

$$\epsilon(\mathbf{n}, d) = \sum_i \widetilde{\delta}\left(\frac{G_i}{h}\right) \times \widetilde{\delta}\left(\frac{\|\mathbf{x}_i - \mathbf{x}_c\|_2}{h}\right) \times (G_i - \mathbf{n} \cdot (\mathbf{x}_i - \mathbf{x}_c) - d)^2, \tag{22}$$

where $\widetilde{\delta}$ is a smoothed Dirac distribution. $\mathbf{x}_c$ is the center of mass of the interfacial element (a local coordinate system is used to reduce the impact of round-off errors). Note that for a given characteristic length h or the order of the grid size, the first factor increases the weight of small $|G|$ values (therefore close to the interface) while the second one excludes points far from the element.

4.2.2. Volume fraction matching

After normalization to ensure $|\mathbf{n}| = 1$, d is iteratively modified to match the volume fraction value ($F = F_{\mathcal{E}}^{n}$ over $\Omega = \Omega_{\mathcal{E}}^{n}$ for the backward projection, $F = F_{\mathcal{L}}^{n+1}$ over $\Omega = \Omega_{\mathcal{L}}^{n+1}$ for the forward projection),

$$\begin{cases} d_0 & = d, \\ d_{k+1} & = d_k + l_k, \end{cases} \tag{23}$$

where

$$\begin{cases} \mathrm{res}_k = F \int_\Omega d\mathbf{x} - \int_\Omega H(\mathbf{n} \cdot (\mathbf{x} - \mathbf{x}_c) + d_k)\, d\mathbf{x}, \\ l_k = \dfrac{\mathrm{res}_k}{\int_\Omega \delta(\mathbf{n} \cdot (\mathbf{x} - \mathbf{x}_c) + d_k)\, d\Omega}, \end{cases} \tag{24}$$

which is used until $|\mathrm{res}_k| \leqslant \mathrm{tol} \int_\Omega d\mathbf{x}$.

Note that the last term in res_k and the denominator in l_k correspond to the liquid volume and interface area corresponding to kth estimate $\mathcal{P}$, respectively. Since Ω can have arbitrary shape but has been triangulated, the Marching Tetrahedra algorithm is used to compute both using the signature of each simplex for the forward projection.

In practice, it was found that on average, 6–7 iterations are necessary to reach $\mathrm{tol} = 10^{-15}$. The maximum number of iterations N_{iter} was therefore set to 10.

4.3. Intersection computation

After computing the signature of the triangulation, the algorithm relies on the computation of the intersection of

- A signed tetrahedron (from the triangulation of a Lagrangian element).
- A half-space (liquid side of the plane interface).
- A convex target element (a right hexahedron for example).

Since all of these polyhedra are convex (but not necessarily bounded), their intersection also is convex. Also note that the sign of the tetrahedron is inherited by the resulting polyhedron (this is important for volume computation).

Any bounded convex polyhedron can be fully described as the convex hull of a finite set of vertices (called $\mathcal{V}$ or parametric representation), or as the intersection of a set of half-spaces ($\mathcal{H}$ or implicit representation). Note, however, that a half-space cannot be described using a $\mathcal{V}$-representation as defined above, since it is not bounded. The reason is that the actual definition of a $\mathcal{V}$-representation should also include rays and lines in addition to vertices ([34] for further details). Both of these representations are related by duality, and going from one representation to the other one is a non-trivial task. Development of algorithms for doing so is an active research area in Computational Geometry and is a problem known to scale as $\mathcal{O}(n^{D/2})$, where n is the number of half-spaces and D the dimension.

Note that given an $\mathcal{H}$-representation of a set of convex polyhedra, it is easy to see that the resulting intersection is fully characterized by the intersection of all half-spaces. The issue, however, is that concatenating the list of all half-spaces does not, in general, result in a minimal representation of the polyhedron, which is necessary if one wants to use a volume computation algorithm such as Lasserre's formula [35].

In this work, given the low dimensionality of the problem, a mixed representation was used, with saturation and connectivity information stored in order to increase the efficiency of the intersection and volume computations. Given a bounded convex polyhedron, which in the present case is either a simplex or a right hexahedron, with the corresponding saturation and connectivity, the algorithm relies on the elementary operation of computing the intersection of the latter polyhedron with a half-space.

4.4. Volume computation

A variety of algorithms exists to compute the volume of a convex polyhedron, based on recursive formulas, triangulation or signed decomposition methods [36].

A simple triangulation technique can be derived as follows: for a convex polyhedron resulting from the intersection computation algorithm presented above, it is easy to triangulate each face using the saturation and connectivity information. Adding the center of mass of the polyhedron to the set of vertices adjacent to each resulting triangle leads to a triangulation of the initial polyhedron. The total volume is obtained by adding up each of the volumes of each tetrahedron. In our case, this was found to be faster and more robust than a signed decomposition method such as Lasserre's recursive formula.

5. Implementation in a structured finite difference solver

The following section presents an implementation of the proposed method in a structured finite difference flow solver. The algorithm is also presented in details here to make the implementation more clear.

5.1. Flow solver

The incompressible form of the Navier–Stokes equations is solved on a structured Cartesian mesh. The equation solved is

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\mathbf{u} = -\frac{1}{\rho}\boldsymbol{\nabla} p + \frac{1}{\rho}\boldsymbol{\nabla}\cdot\left(\mu\left[\boldsymbol{\nabla}\mathbf{u} + \boldsymbol{\nabla}\mathbf{u}^t\right]\right) + \mathbf{g}, \tag{25}$$

where $\mathbf{u}$ is the flow velocity, ρ is the density, p is the pressure, μ is the dynamic viscosity, and $\mathbf{g}$ represents the body forces.

The high-order conservative finite difference scheme described in [37] is used. Spatial and time accuracy are limited to second order, which is consistent with the Lagrangian–Eulerian Volume-of-Fluid method proposed. A fractional step approach is used: the body force, convective and viscous terms are advanced using semi-implicit Newton-Raphson iterations combined with an ADI factorization for efficient time advancement. A projection step following the Ghost Fluid Method [38] ensures both the divergence free condition,

$$\boldsymbol{\nabla}\cdot\mathbf{u} = 0, \tag{26}$$

and the pressure jump condition,

$$[p]_\Gamma = \sigma\kappa + [\mu]_\Gamma \mathbf{n}^t\cdot\boldsymbol{\nabla}\mathbf{u}\cdot\mathbf{n}. \tag{27}$$

It also allows for a discrete balance of pressure and surface tension forces. A harmonic averaging is used to treat the viscous term [8]. In Eq. (27), $\mathbf{n}$ refers to the interface normal vector obtained from the Level Set field, and the curvature is computed using a discretization of

$$\kappa = -\boldsymbol{\nabla}\cdot\mathbf{n}. \tag{28}$$

Note that in the context of Volume-of-Fluid methods, a height function technique [25–29] would most likely yield a better curvature estimate.

The velocity components are staggered in space, and stored at time n, u_1, u_2, and u_3 are located at cell faces $(i+1/2, j, k)$, $(i, j+1/2, k)$, and $(i+1/2, j, k+1/2)$, respectively.

In addition, the convective CFL number is limited to unity to limit the neighbor search in the remapping step of the Volume-of-Fluid transport.

5.2. Level Set

The Level Set field is transported as follows (details in [37]): G is stored at cell centers at time (n), and the divergence operator used in Eq. (26) is also used to solve 9. A High Order Upwind Central scheme is used to evaluate the fluxes at cell faces (Newton-Raphson iterations are also used).

5.3. Volume-of-Fluid

Similar to the Level Set field G, the volume fraction is defined over the control volumes centered at (i,j,k) at time n.

An additional subtlety which was found to affect the conservation properties is the benefit of combining the Forward and Backward projections. It will be pointed out that the mass conservation properties of both projections are negatively correlated: for a given velocity field, if the use of a Backward projection step leads to mass loss, the Forward projection leads to mass gain, and vice versa. Consequently, this suggests that combining both projections to form a trapezoidal rule type scheme may lead to better mass conservation properties.

Lastly, it has to be noted that even for a simple linear velocity field, the accuracy to which the vertex transport Eqs. (12) and (15) are solved has a strong impact on the geometric accuracy and the conservation properties of the proposed method. Various explicit Runge-Kutta methods were therefore tested, which may be recast for the backward projection as follows,

$$\begin{cases} \mathbf{x}^n = \mathbf{x}^{n+1} - \Delta t \sum_{i=1}^{s} b_i \mathbf{k}_i, \\ \mathbf{k}_i = \widehat{\mathbf{u}}\left(t^{n+1} - c_i \Delta t, \mathbf{x}^{n+1} - \Delta t \sum_{j=1}^{s} a_{ij} \mathbf{k}_j \right), \end{cases} \tag{29}$$

where the coefficients are given in a Butcher tableau of the form,

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1s} \\ c_2 & a_{21} & a_{22} & \dots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s2} & \dots & a_{ss} \\ \hline & b_1 & b_2 & \dots & b_s \end{array} . \tag{30}$$

The key component in formula (29) is how the velocity approximation $\widehat{\mathbf{u}}$ is reconstructed, which needs to be continuous. In addition, for linear velocity fields for example, it should also be exact for the volume fraction transport to be second order accurate. Both requirements are easily met on a staggered mesh by combining a linear interpolation in time and a trilinear interpolation in space for each of the velocity components.

Note that this reconstructed velocity field, while continuous, is not divergence free. The error resulting conservation error however is only third order.

5.4. Algorithm details

The pseudo-code presented in this section provides a detailed description of the HyLEM method in the structured finite difference framework presented above.

5.4.1. Time-step structure

This first algorithm presents the structure of a single time step including the main steps, which are also detailed below. It includes the Newton-Raphson iteration index k.

Algorithm 1. Trapezoidal rule time advancement template.

Require: $G^n, F^n, \mathbf{u}^n$
  Compute Δt such that $\mathrm{CFL}_{conv} \leqslant 1$
  $G_0^{n+1} \leftarrow G^n$
  $\mathbf{u}_0^{n+1} \leftarrow \mathbf{u}^n$
  $\mathcal{E}_\Gamma^n \leftarrow$ **tag_interface**(F^n)
  for $k = 1, N_{iter}$ **do**
    $G_{k+1}^{n+1} \leftarrow$ **solve_levelset**$\left(G^n, G_k^{n+1}, \mathbf{u}^n, \mathbf{u}_k^{n+1}\right)$
    $F_{forw}^{n+1} \leftarrow$ **forward_projection**$\left(G_{k+1}^{n+1}, F^n, +\mathbf{u}^n \Delta t, +\mathbf{u}_k^{n+1} \Delta t, \mathcal{E}_\Gamma^n\right)$
    $F_{back}^{n+1} \leftarrow$ **backward_projection**$\left(G^n, F^n, -\mathbf{u}^n \Delta t, -\mathbf{u}_k^{n+1} \Delta t, \mathcal{E}_\Gamma^n\right)$
    $F_{k+1}^{n+1} \leftarrow 1/2 \times \left(F_{forw}^{n+1} + F_{back}^{n+1}\right)$
    $G_{k+1}^{n+1} \leftarrow$ **correct_and_reinit**$\left(F_{k+1}^{n+1}, G_{k+1}^{n+1}\right)$
    $\mathbf{u}_{k+1}^{n+1} \leftarrow$ **solve_flow**$\left(G^n, G_{k+1}^{n+1}, F^n, F_{k+1}^{n+1}, \mathbf{u}^n, \mathbf{u}_k^{n+1}\right)$.
  end for

Since it is slightly more complex than a single forward or single backward projection, the trapezoidal rule scheme, which results from the arithmetic average of both projections, is presented.

5.4.2. Interface tagging

In Algorithm 1, the notation $\mathcal{E}_\Gamma^n$ was used to denote a subset of elements from the target mesh $\mathcal{E}^n$. The major cost in the HyLEM method comes from the intersection algorithm. It is therefore crucial to bring the number of calls to these routines to a minimum. As pictured in Fig. 10, only the Eulerian cells neighboring the interface are likely to see their volume fraction change, assuming a convective CFL number lower than unity.

Note again that this restriction is not necessary: the Lagrangian–Eulerian framework described so far does not forbid higher CFL number, but allowing it to exceed unity implies increasing the number of "reachable cells". In 3D, for $\mathrm{CFL}_{conv} \leqslant 1$, a given interfacial cell has 27 neighbors, and this number goes up to 125 for $\mathrm{CFL}_{conv} \leqslant 2$. Since the stability of the flow solver is usually not guaranteed for convective CFL number higher than unity, this does not constrain the computational time step.

In addition, the following strategy also reduces the number of polyhedral operations. Initialize the volume fraction field as,

$$F_0^{n+1} = \begin{cases} 0 & \text{if } F^n < 0.5, \\ 1 & \text{if } F^n \geqslant 0.5, \end{cases} \tag{31}$$

and assign levels to every interfacial cell as shown in Fig. 10.

One simply needs to add (resp. subtract) the intersecting liquid (resp. gas) fraction to any cell initially satisfying $F^{n+1} = 0$ (resp. $F^{n+1} = 1$). From this statement, a few simplifications follow. For instance, one can easily see that for a forward projection, there is no need to compute the intersection of a cell with level +2 with a cell of level +1.

5.4.3. Projection steps

Algorithms 2 and 3 provide a clear description of how the forward and backward projection steps may be performed.

Algorithm 2. Forward projection time advancement template.

Require: $G^{n+1}, F^n, +\mathbf{u}^n\Delta t, +\mathbf{u}^{n+1}\Delta t$ and $\mathcal{E}_\Gamma^n$.
  for cells in $\mathcal{E}_\Gamma^n$ **do**
    advance each vertex forward in time using $+\mathbf{u}^n\Delta t$ and $+\mathbf{u}^{n+1}\Delta t$.
    determine sign for each tetrahedron of $\Omega_\mathcal{L}^{n+1}$.
    guess interface plane $\left(\mathbf{n}_\mathcal{L}^{n+1}, d_\mathcal{L}^{n+1}\right)$ from G^{n+1}.
    modify $d_\mathcal{L}^{n+1}$ to match $F_\mathcal{L}^{n+1} = F_\mathcal{E}^n$.
    intersect $F_\mathcal{L}^{n+1}$ with neighbors $\Omega_\mathcal{E}^{n+1}$ and update F_{forw}^{n+1}.
  end for

Algorithm 3. Backward projection time advancement template.

Require: $G^n, F^n, -\mathbf{u}^n\Delta t, -\mathbf{u}^{n+1}\Delta t$ and $\mathcal{E}_\Gamma^n$.
  for cells in $\mathcal{E}_\Gamma^n$ **do**
    guess interface plane $(\mathbf{n}^n, d^n)$ from G^n.
    modify d^n to match $F_\mathcal{E}^n$.
  end for
  for cells in $\mathcal{E}_\Gamma^n$ **do**
    advance each vertex backward in time using $-\mathbf{u}^n\Delta t$ and $-\mathbf{u}^{n+1}\Delta t$.
    determine sign for each tetrahedron of $\Omega_\mathcal{L}^n$.
    intersect $\Omega_\mathcal{L}^n$ with neighbors $F_\mathcal{E}$ and update F_{back}^{n+1}.
  end for

6. Conservation and geometrical properties

In this section, the proposed method is proven to conserve mass exactly for linear velocity profiles provided the truncation error in transporting the vertices is small enough. An analytical proof is derived by computing the volume error resulting from the transport of a target element. In addition, the mass error is numerically shown to be third order by means of common transport tests.

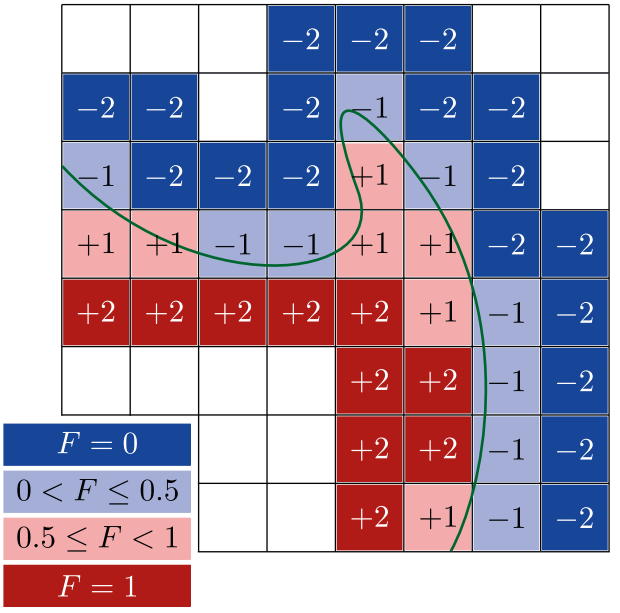


Fig. 10. Levels of active cells in remapping phase.

6.1. Error cancellation for linear velocity field

The conservation error introduced in the forward projection is related to the following error

$$\epsilon = \frac{\mathrm{Vol}_{\mathcal{L}}^{n+1} - \mathrm{Vol}_{\mathcal{E}}^{n}}{\mathrm{Vol}_{\mathcal{E}}^{n}}. \tag{32}$$

This error is now proven to null for a linear divergence free velocity field.

Consider a D-dimensional problem ($D = 1, 2$ or 3) and $N \in \mathbb{N}^{+*}$, and define

- $(g_i : \mathbb{R} \to \mathbb{R} | i = 1, \ldots, N)$ an N-uple of integrable function with respective primitives $(G_i | i = 1, \ldots, N)$.
- $(\mathbf{A}_i | i = 1, \ldots, N) \in \left(\mathbb{R}^D \times \mathbb{R}^D\right)^N$ such that $\forall i = 1, \ldots, N, \mathrm{tr}(\mathbf{A}_i) = 0$ and A_i is diagonalizable ($\mathbf{A}_i = \mathbf{S}_i \mathbf{\Lambda}_i \mathbf{S}_i^{-1}$).
- $(\mathbf{b}_i | i = 1, \ldots, N) \in \left(\mathbb{R}^D\right)^N$.

Define the following velocity field

$$\mathbf{u}(\mathbf{x}, t) = \sum_i (\mathbf{A}_i \cdot \mathbf{x} + \mathbf{b}_i) g_i(t), \tag{33}$$

which can easily be seen to be solenoidal.

The solution of the ordinary differential equation (12) with velocity field (33) is

$$\mathbf{x}(t) = \exp\left(\sum_i \mathbf{A}_i G_i(t)\right) \mathbf{x}_{\mathcal{E}}^n + \int_{t'=t^n}^{t} \exp\left(\sum_i \mathbf{A}_i (G_i(t) - G_i(t'))\right) \left(\sum_i \mathbf{b}_i g_i(t')\right) dt'. \tag{34}$$

Because Eq. (34) is linear, two important facts to note are as follows: first, the image of any convex set of points is also convex, so the image of a convex face will also be a convex face. This is quite reassuring, since so far the current algorithm should not fail.

In addition, the second term in 34 does not depend on $\mathbf{x}_{\mathcal{E}}^n$; it simply translates the target mesh $\mathcal{E}^n$. The volume error ϵ is therefore solely affected by the first term, which is linear in $\mathbf{x}_{\mathcal{E}}^n$ and reads

$$\begin{aligned} \frac{\mathrm{Vol}_{\mathcal{L}}^{n+1}}{\mathrm{Vol}_{\mathcal{E}}^{n}} &= \det\left(\exp\left(\sum_i G_i(t)\mathbf{A}_i\right)\right) = \Pi_i \det\left(\exp\left(G_i(t)\mathbf{A}_i\right)\right), = \Pi_i \det\left(\mathbf{S}_i \exp\left(G_i(t)\mathbf{\Lambda}_i\right)\mathbf{S}_i^{-1}\right) \\ &= \Pi_i \det\left(\exp\left(G_i(t)\mathrm{tr}\mathbf{\Lambda}_i\right)\right), \end{aligned} \tag{35}$$

but $\mathrm{tr}\mathbf{A}_i = \mathrm{tr}\mathbf{\Lambda}_i = 0$, so this term is unity, leading to $\epsilon = 0$.

Provided that Eq. (12) is solved exactly, the current method can therefore conserve mass exactly. Since linear interpolation is used in both space and time, if the time discretization of Eq. (12) is accurate enough, mass may be conserved exactly

for any velocity field linear in space and time. If the truncation error is sufficiently small, high order Runge-Kutta methods can therefore lead to exact mass conservation for linear velocity field.

The flow solver used here is second order, which is consistent with this second order conservation property. It is worthy to note however that for real applications, the velocity field is unlikely to be linear. This motivates the analysis of the conservation error for various cases presented in the following section.

6.2. Transport tests

In order to assess the capabilities of the proposed method, standard interface transport test cases are studied. The errors are analyzed by means of the discrete equivalent of the following estimates: the relative conservation error integrated over the computational domain (CD) is defined as,

$$\epsilon_{\text{mass}} = \frac{\int_{\text{CD}} f(\mathbf{x},t)\,d\mathbf{x} - \int_{\text{CD}} f(\mathbf{x},0)\,d\mathbf{x}}{\int_{\text{CD}} f(\mathbf{x},0)\,d\mathbf{x}}, \tag{36}$$

and for T-periodic cases, the relative geometric error reads,

$$\epsilon_{\text{geo}} = \frac{\int_{\text{CD}} |f(\mathbf{x},T) - f(\mathbf{x},0)|\,d\mathbf{x}}{\int_{\text{CD}} f(\mathbf{x},0)\,d\mathbf{x}}. \tag{37}$$

6.2.1. Zalesak's disk

Zalesak's disk is a two-dimensional test cased used to characterize how accurately sharp corners are transported. A solid body rotation velocity field centered on $(0,0)$ is used to transport the interface, which is initialized as a notched disk centered at $(0,0.25)$. The circle radius is 0.15, and the notch is 0.05×0.25. All cases presented were run with a unity CFL number.

The conservation errors plotted in Fig. 11 illustrate two interesting features. First, as suggested from the analytical proof provided in Section 6.1, the conservation error is expected to tend quickly to zero as the grid spacing decreases and the time accuracy of the increases. A fast rate of convergence is indeed observed, but it has to be pointed out that this is a simple consequence of the ability of the proposed method to conserve mass exactly for linear velocity fields. This suggests that higher order RK schemes are necessary for the vertex transport. Therefore, all the computations presented in the following sections are performed using a fourth order RK scheme.

In addition, the conservation errors of the backward and forward projection were found to be negatively correlated. Consequently, their combination (trapezoidal rule) was found to dramatically improve the conservation properties.

The combination of these two features resulted in close to machine precision conservation error for all grids using the trapezoidal rule projection with a fourth order vertex transport. This also illustrates the robustness and accuracy of the polyhedral library implemented for this study.

Except for the first order forward and backward projections, the geometric accuracy on the other hand was found to be close to almost constant, as seen in Table 1. Hence, no difference could be seen in Fig. 12(a)–(c) between the different pro-

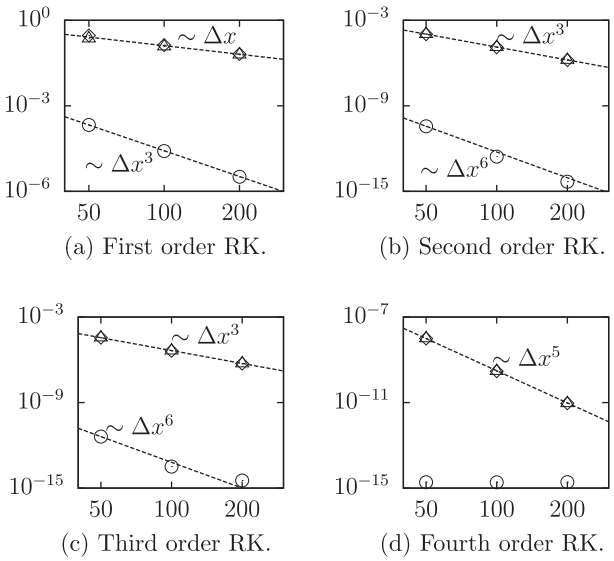


Fig. 11. Effect of truncation error in vertex transport on ϵ_{mass} for Zalesak's test (Backward Euler: *triangles*, Forward Euler: *diamonds*, Trapezoidal rule: *circles*).

Table 1
Geometric error ϵ_{geo} for Zalesak's disk on a 200×200 grid after one complete revolution.

Projection type	RK1	RK2	RK3	RK4
Backward Euler	$1.11 \cdot 10^{-1}$	$9.04 \cdot 10^{-3}$	$8.93 \cdot 10^{-3}$	$8.93 \cdot 10^{-3}$
Forward Euler	$1.19 \cdot 10^{-1}$	$9.76 \cdot 10^{-3}$	$9.70 \cdot 10^{-3}$	$9.70 \cdot 10^{-3}$
Trapezoidal rule	$9.75 \cdot 10^{-3}$	$9.39 \cdot 10^{-3}$	$9.31 \cdot 10^{-3}$	$9.31 \cdot 10^{-3}$

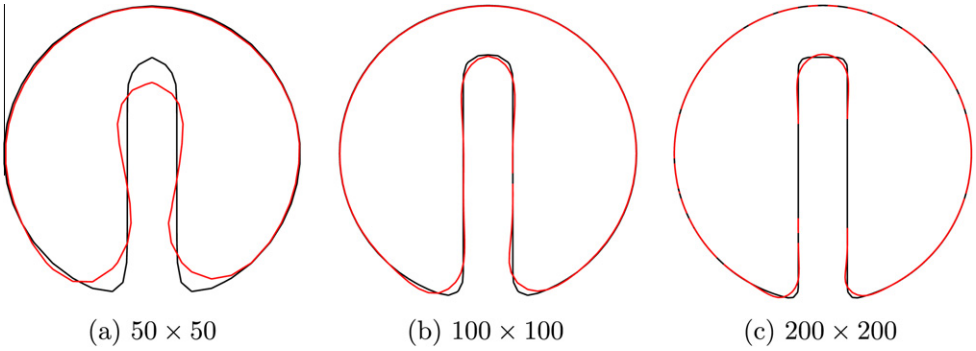


Fig. 12. Zalesak's disk for various grid sizes (RK4 used, initial: *black*, after complete revolution: *red*). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

jections. In addition, the reported values are consistent with other volume-of-fluid methods relying on piecewise linear reconstruction [39].

6.2.2. Disk in deformation field

This case tests the ability of the proposed method to robustly represent under-resolved structures such as ligaments. The following stream function is prescribed over a $[0,1] \times [0,1]$ domain,

$$\phi(\mathbf{x},t) = \frac{1}{\pi} \sin^2(\pi x) \cos^2(\pi y) \cos\left(\frac{\pi t}{T}\right), \tag{38}$$

where $T = 8$. The geometry is initialized as a disk of radius 0.15 centered at $(0.5, 0.75)$. All cases were run with a CFL number of 0.5.

Unlike Zalesak's disk, the prescribed velocity field is not linear, which makes this test case more realistic. The three proposed projection schemes are tested, and Fig. 13 reports the relative conservation error. The vertex transport is performed using a fourth order Runge-Kutta method.

The convergence rate for both forward and backward projections was found to be third order. The error levels are comparable to dimensional splitting methods, such as the Coupled Level Set and Volume-of-Fluid method (in [7], the authors report an error of the order of 0.01% for a 128×128 grid).

The trapezoidal rule benefits from the negatively correlated errors of both Forward and Backward projections, as already seen in Zalesak's test case, which leads to error levels smaller by four or more orders of magnitude. The convergence rate quickly degrades as the conservation error becomes close to machine precision accuracy.

Fig. 14 shows the interface position at $T/2$ for various grids, along with a reference solution obtained on a 1024×1024 grid. The algorithm clearly proves to be robust in smooth regions, as seen from the good agreement in all regions but the tips of the ligament where the curvature radius becomes close to the grid spacing. The final geometries are shown on Fig. 15.

6.2.3. Sphere in deformation field

This test case was proposed in [13] following [40] and is used here to illustrate the robustness and convergence properties in a three-dimensional configuration. The interface is transported with the following velocity field,

$$\begin{cases} u(x,y,z,t) = 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)\cos(\pi t/T), \\ v(x,y,z,t) = -\sin(2\pi x)\sin^2(\pi y)\sin(2\pi z)\cos(\pi t/T), \\ w(x,y,z,t) = -\sin(2\pi x)\sin(2\pi y)\sin^2(\pi z)\cos(\pi t/T), \end{cases} \tag{39}$$

with $T = 3$. The interface is initialized as a spherical drop of radius 0.15 centered at $(0.35, 0.35, 0.35)$ over a $[0,1] \times [0,1] \times [0,1]$ domain. The domain is discretized using 192 equally spaced points along each direction, the CFL number was set to 0.5 and the vertex transport scheme is a fourth order RK.

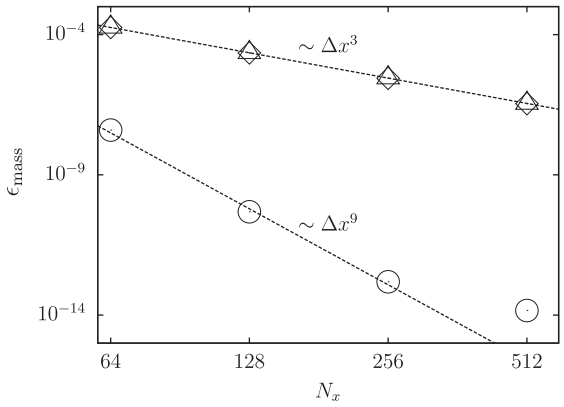


Fig. 13. Conservation error as a function of grid size for disk in 2D deformation case (RK4 used, Backward Euler: *triangles*, Forward Euler: *diamonds*, Trapezoidal rule: *circles*).

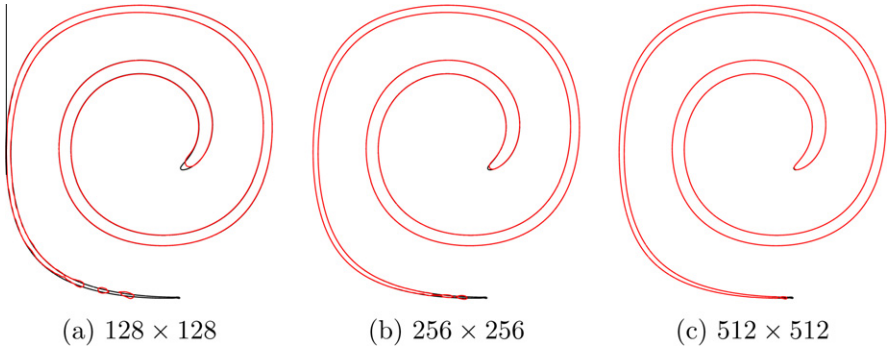


Fig. 14. Interface position at $t = T/2$ for 2D deformation case (HyLEM with RK4: *red*, reference: *black*). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

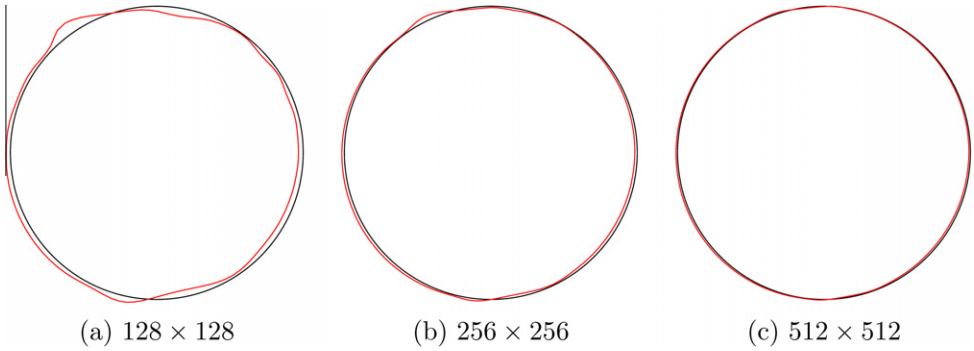


Fig. 15. Interface position at $t = T$ for 2D deformation case (HyLEM with RK4: *red*, exact: *black*). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

As seen in Fig. 16 where a cut of the interface shown in Fig. 17(a) is illustrated, the interface is resolved over a single cell and the proposed algorithm is able to maintain a proper representation of the interface.

At final time $t = T$, the shape obtained is close to the initial sphere, except for a slight deformation on which Fig. 17(b) is centered. In addition, mass conservation error remains very low as seen from Fig. 18, and the third order convergence rate of the Forward and Backward transports is again clearly seen.

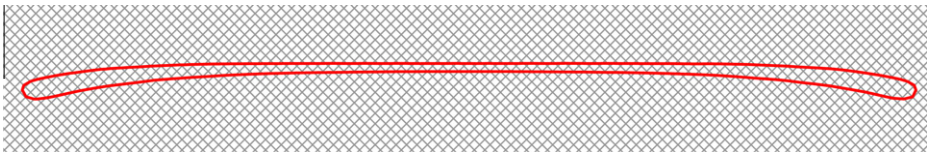


Fig. 16. Two-dimensional cut at maximum stretching ($t = T/2$) for 3D deformation on 192^3 grid.

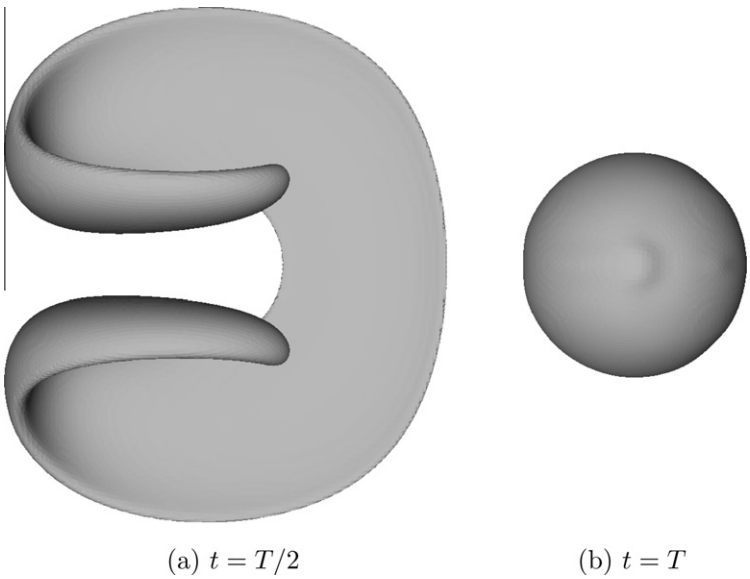


Fig. 17. Interface position for 3D deformation case on a 192^3 grid (RK4).

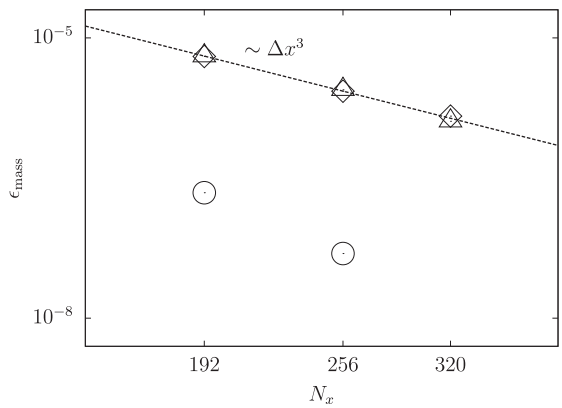


Fig. 18. Conservation error as a function of grid size for disk in 3D deformation case (Backward Euler: *triangles*, Forward Euler: *diamonds*, Trapezoidal rule: *circles*).

7. Applications

This section aims mainly at illustrating the capabilities of the proposed method for computations involving complex topologies.

7.1. Standing wave

This test case aims at validating the proposed relaxation formula (16). Based on the work presented in [41], it consists in comparing the analytical decay of a two-dimensional disturbed interface initialized in a $[-\pi,\pi]\times[-\pi,\pi]$ domain as the zero iso-contour of,

$$\varphi(x,y)=\pi-y+A_0\cos\left(\frac{2\pi x}{\lambda}\right), \tag{40}$$

to a numerical solution (λ is set to $2\pi, A_0$ to 0.01λ).

Following [8], the surface tension coefficient is set to $\sigma=2$, the density in fluid 1 is $\rho_1=1$, and two different cases are studied. The first case uses ρ_2 to 1.0 with identical kinematic viscosity $\nu=0.064720863$ in both fluids, while in the second case ρ_2 is set to 1000.0 and the kinematic viscosity to $\nu=0.0064720863$. The time-step is set to 0.003 in the first case, 0.06 in the second, and in both cases a fourth order RK scheme is used for the vertex transport.

The amplitude profiles are reported for different meshes in Figs. 19(a) and 20(a) for the first and second case respectively, and the relative errors in Figs. 19(b) and 20(b). In addition, the RMS of the relative error are reported for both cases in Table 2. It can be seen that for the unity density ratio, results are comparable to the Level Set results presented in [8]. For the high density ratio case however, it is interesting to note that the proposed method performs better.

7.2. Rayleigh–Taylor

The HyLEM method is employed to study the growth of a two-dimensional Rayleigh–Taylor instability. This test case follows the computational study originally presented in [42] and later in [8,14].

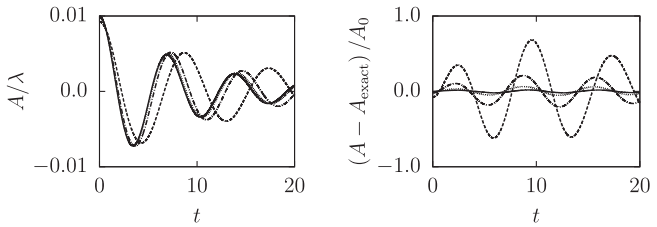
The interface is initialized as the zero iso-contour of the following Level Set field,

$$G_0(x,y)=y+A\cos(2\pi x), \tag{41}$$

over a $[0,1]\times[0,4]$ domain with the perturbation amplitude $A=0.05$. The physical properties are taken as $\rho_1=1.225$ and $\mu_1=0.00313$ for the heavy fluid, $\rho_2=0.1694$ and $\mu_2=\mu_1$ for the light fluid and $\sigma=0.1337$ for the surface tension. Following [8], the time-step is set to 2.5×10^{-4}.

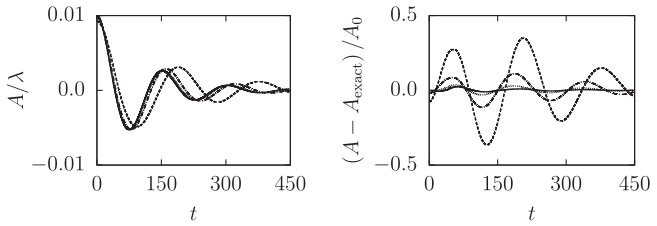
The computation is carried on grids ranging from 32×128 to 512×2048, the time-step is taken as half of the capillary time-step on the finest grid, and the vertex transport performed with a fourth order RK scheme. The results on the finest grid are used as the reference solution.

The results shown in Fig. 21 are in good agreement with those reported in [42,14]. At time $t=1.0$, only the two coarsest grids slightly deviate from the reference solution obtained on the 512×2048 grid, while at time $t=1.1$ there is still no vis-



(a) Disturbance amplitude. (b) Relative error.

Fig. 19. Damped surface wave case with unity density contrast: 8×8 mesh (dashed line), 16×16 mesh (dash-dotted line), 32×32 (dotted line), 64×64 (solid line).



(a) Disturbance amplitude. (b) Relative error.

Fig. 20. Damped surface wave case with $1:1000$ density contrast: 8×8 mesh (dashed line), 16×16 mesh (dash-dotted line), 32×32 (dotted line), 64×64 (solid line).

Table 2
RMS of the relative error in standing wave case.

Mesh size	$\rho_2/\rho_1 = 1$	$\rho_2/\rho_1 = 1000$
8×8	0.3844	0.1894
16×16	0.1277	0.0587
32×32	0.0371	0.0164
64×64	0.0125	0.0074

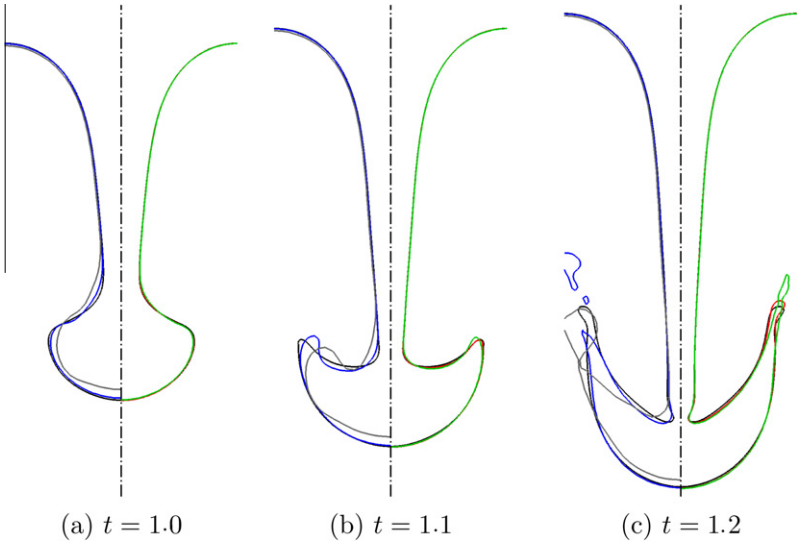


Fig. 21. Interface location for the Rayleigh–Taylor instability problem (RK4 used, 32×128: *gray*, 64×256: *blue*, 128×512: *green*, 256×1024: *red*, 512×2048: *black*). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

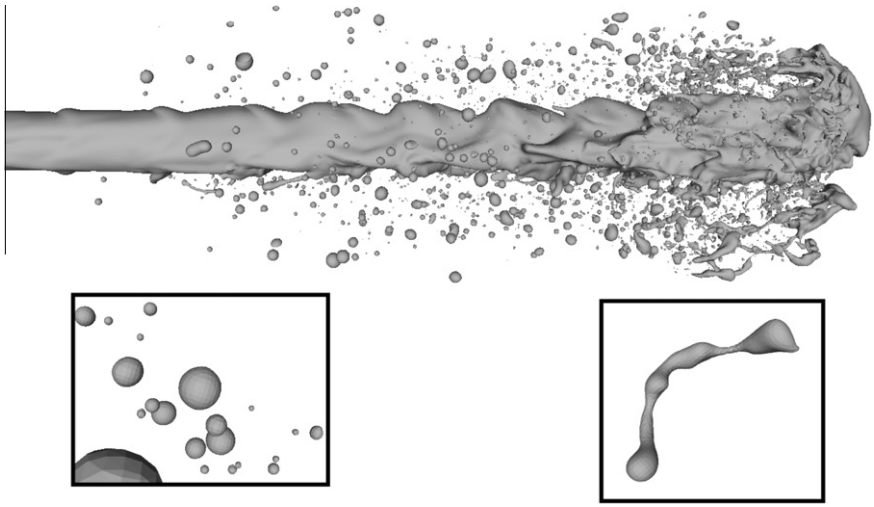


Fig. 22. Interface position at $t = 15U_{\text{bulk}}/D$ (RK4 used).

ible difference between the two finest grids. At time $t = 1.2$ however, as the curvature radius increases at the edges of the mushroom cap, the convergence rate deteriorates as expected from Volume-of-Fluid methods in under-resolved regions of the flow.

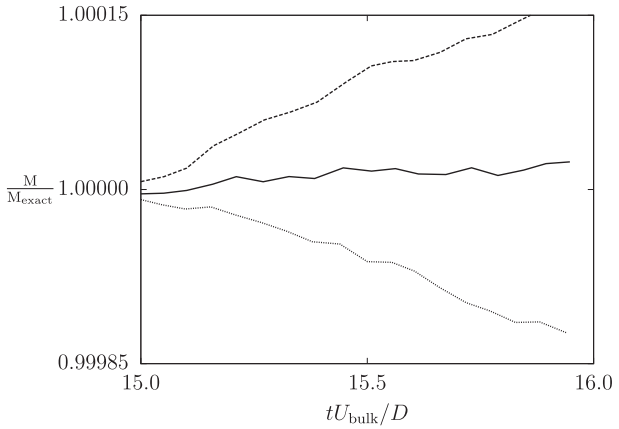


Fig. 23. Comparison of mass errors in diesel jet (RK4 used, *dashed*: Backward Euler, *dotted*: Forward Euler, *solid*: Trapezoidal rule).

Table 3
Averaged computational budget in diesel jet computation (excluding I/O). Averaged time per iteration: 22.3 s on 720 processors.

Solver	Relative cost (%)
Pressure	62.2
Multiphase	23.5
Velocity	11.8
Rest	2.5

7.3. Diesel jet

In order to validate the proposed method for complex interface topology, a spatially evolving round jet is computed. The density ratio used is 20, the jet Reynolds number $Re = 1000$ in both phases and the liquid Weber number based on the jet diameter $We = 1000$. A $256 \times 256 \times 1152$ mesh is used, spanning a $4D \times 4D \times 18D$ domain, the time-step was set so as for the convective CFL number not to exceed 0.8, and the vertex transport was performed with a fourth order RK scheme. The objective here is not to study the underlying atomization process, but rather to provide a realistic interface topology to compare the various time advancements proposed in this work.

The interface is shown at $t = 15U_{bulk}/D$ in Fig. 22, along with a cloud of droplets in a dilute region of the flow. It can be seen that three to four points are enough to resolve a droplet. In addition, a ligament is also represented, with a curvature radius locally comparable to the grid spacing.

In order to evaluate the performances of the various time advancement schemes, the configuration shown in Fig. 22 is used as initial condition, and the flow is advanced for one characteristic time scale U_{bulk}/D. The relative mass errors are shown in Fig. 23.

As expected, the errors for the Forward and Backward schemes are negatively correlated. The error levels are relatively low (of the order of 0.015%), and the combination of both is seen to reduce the mass loss by an order of magnitude (of the order of 0.002% after one characteristic time scale).

This case is also used to quantify the computational cost associated with the polyhedral operations. As seen in Table 3, the efficient implementation described in the previous sections achieves reasonable performances, the interface solver representing less than one fourth of the total cost.

8. Conclusion

A coupled unsplit Volume-of-Fluid and Level Set method for the computation of free surface flows has been proposed. The flexibility inherent to the proposed geometric framework resulted in the development of various time advancement schemes. The properties of these algorithms have been analytically and numerically studied in 2D and 3D computations.

In particular, it has been shown that the HyLEM method could conserve mass exactly for linear velocity fields provided the truncation error in the mesh deformation was small enough. This property and the ability of the method to represent planar interfaces exactly makes the proposed method formally second order.

For non-linear velocity fields, the conservation error was shown to be very small, even compared with split Volume-of-Fluid methods, while the convergence rate for the mass conservation error was shown to be third order. This property is a direct consequence of the unsplit nature of the Volume-of-Fluid transport.

Finally, the method was applied to a round jet computation, and the mass conservation error was shown to remain satisfactory despite the complex topology, which illustrates the ability of the proposed method to accurately compute realistic configurations.

Acknowledgments

The authors wish to express their gratitude to Dr. Ed Knudsen for his comments on a draft of this manuscript. We also gratefully acknowledge funding by NASA.

References

[1] A.A. Mostafa, H.C. Mongia, On the modeling of turbulent evaporating sprays: Eulerian versus Lagrangian approach, International Journal of Heat and Mass Transfer 30 (1987) 2583–2593.
[2] M.G. Pai, S. Subramaniam, A comprehensive probability density function formalism for multiphase flows, Journal of Fluid Mechanics 628 (2009) 181–228.
[3] J.K. Dukowicz, A particle–fluid numerical model for liquid sprays, Journal of Computational Physics 35 (1980) 229–253.
[4] R.S. Miller, J. Bellan, Direct numerical simulation of a confined three-dimensional gas mixing layer with one evaporating hydrocarbon-droplet-laden stream, Journal of Fluid Mechanics 384 (1999) 293–338.
[5] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Annual Review of Fluid Mechanics 31 (1999) 567–603.
[6] G. Tryggvason, R. Scardovelli, S. Zaleski, Direct Numerical Simulations of Gas–liquid Multiphase Flows, Cambridge University Press, Cambridge, New York, 2011.
[7] T. Menard, S. Tanguy, A. Berlemont, Coupling level set/VOF/ghost fluid methods: validation and application to 3D simulation of the primary break-up of a liquid jet, International Journal of Multiphase Flow 33 (2007) 510–524.
[8] M. Herrmann, A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids, Journal of Computational Physics 227 (2008) 2674–2706.
[9] S.O. Unverdi, G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, Journal of Computational Physics 100 (1992) 25–37.
[10] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, Journal of Computational Physics 114 (1994) 146–159.
[11] J.A. Sethian, Evolution, implementation, and application of level set and fast marching methods for advancing fronts, Journal of Computational Physics 169 (2001) 503–555.
[12] E. Marchandise, J.-F. Remacle, N. Chevaugeon, A quadrature-free discontinuous Galerkin method for the level set equation, Journal of Computational Physics 212 (2006) 338–357.
[13] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, Journal of Computational Physics 183 (2002) 83–116.
[14] O. Desjardins, H. Pitsch, A spectrally refined interface approach for simulating multiphase flows, Journal of Computational Physics 228 (2009) 1658–1677.
[15] W.J. Rider, D.B. Kothe, Reconstructing volume tracking, Journal of Computational Physics 141 (1998) 112–152.
[16] P. Liovic, M. Rudman, J.-L. Liow, D. Lakehal, D. Kothe, A 3D unsplit-advection volume tracking algorithm with planarity-preserving interface reconstruction, Computers and Fluids 35 (2006) 1011–1032.
[17] M. Rudman, A volume-tracking method for incompressible multifluid flows with large density variations, International Journal for Numerical Methods in Fluids 28 (1998) 357–378.
[18] G.D. Weymouth, D.K.P. Yue, Conservative volume-of-fluid method for free-surface simulations on cartesian-grids, Journal of Computational Physics 229 (2010) 2853–2865.
[19] Y. Renardy, M. Renardy, Prost: a parabolic reconstruction of surface tension for the volume-of-fluid method, Journal of Computational Physics 183 (2002) 400–421.
[20] Y. Morinishi, Skew-symmetric form of convective terms and fully conservative finite difference schemes for variable density low-mach number flows, Journal of Computational Physics 229 (2010) 276–300.
[21] E. Olsson, G. Kreiss, A conservative level set method for two phase flow, Journal of Computational Physics 210 (2005) 225–246.
[22] O. Desjardins, V. Moureau, H. Pitsch, An accurate conservative level set/ghost fluid method for simulating turbulent atomization, Journal of Computational Physics 227 (2008) 8395–8416.
[23] A.J. Chorin, Curvature and solidification, Journal of Computational Physics 57 (1985) 472–490.
[24] B.D. Nichols, C.W. Hirt, R.S. Hotchkiss, Sola-VOF: A Solution Algorithm for Transient Fluid Flow with Multiple Free Boundaries, Technical Report UCRL-ID-126402, Los Alamos National Laboratory, 1980.
[25] S. Popinet, An accurate adaptive solver for surface-tension-driven interfacial flows, Journal of Computational Physics 228 (2009) 5838–5866.
[26] S. Afkhami, M. Bussmann, Height functions for applying contact angles to 3D VOF simulations, International Journal for Numerical Methods in Fluids 61 (2009) 827–847.
[27] M.M. Francois, S.J. Cummins, E.D. Dendy, D.B. Kothe, J.M. Sicilian, M.W. Williams, A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework, Journal of Computational Physics 213 (2006) 141–173.
[28] E.G. Puckett, On the second-order accuracy of volume-of-fluid interface reconstruction algorithms: convergence in the max norm, Communications in Applied Mathematics and Computational Science 5 (2010) 99–148.
[29] M. Sussman, A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles, Journal of Computational Physics 187 (2003) 110–136.
[30] M. Sussman, E.G. Puckett, A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows, Journal of Computational Physics 162 (2000) 301–337.
[31] X. Yang, A.J. James, J. Lowengrub, X. Zheng, V. Cristini, An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids, Journal of Computational Physics 217 (2006) 364–394.
[32] J.K. Dukowicz, J.W. Kodis, Accurate conservative remapping (rezoning) for arbitrary Lagrangian–Eulerian computations, SIAM Journal on Scientific and Statistical Computing 8 (1987) 305–321.
[33] K. Shahbazi, M. Paraschivoiu, J. Mostaghimi, Second order accurate volume tracking based on remapping for triangular meshes, Journal of Computational Physics 188 (2003) 100–122.
[34] D. Wilde, A Library for Doing Polyhedral Operations, 1993.

[35] J.B. Lasserre, An analytical expression and an algorithm for the volume of a convex polyhedron in n-dimensions, Journal of Optimization Theory and Applications 39 (1983) 363–377.
[36] A. Enge, K. Fukuda, Exact volume computation for polytopes: a practical study, Polytopes: Combinatorics and Computation (1997).
[37] O. Desjardins, G. Blanquart, G. Balarac, H. Pitsch, High order conservative finite difference scheme for variable density low mach number turbulent flows, Journal of Computational Physics 227 (2008) 7125–7159.
[38] R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), Journal of Computational Physics 152 (1999) 457–492.
[39] J. Lopez, J. Hernandez, P. Gomez, F. Faura, A volume of fluid method based on multidimensional advection and spline interface reconstruction, Journal of Computational Physics 195 (2004) 718–742.
[40] R.J. Leveque, High-resolution conservative algorithms for advection in incompressible flow, SIAM Journal on Numerical Analysis 33 (1996) 627–665.
[41] A. Prosperetti, Motion of two superposed viscous fluids, Physics of Fluids 24 (1981) 1217–1223.
[42] P. Gomez, J. Hernandez, J. Lopez, On the reinitialization procedure in a narrow-band locally refined level set method for interfacial flows, International Journal for Numerical Methods in Engineering 63 (2005) 1478–1512.