University of Groningen

A comparison between the surface compression method and an interface reconstruction method for the VOF approach

Cifani, Paolo; Michalek, W. R. ; Priems, G. J. M.; Kuerten, J. G. M.; van der Geld, C. W. M.; Geurts, B. J.

Published in:
Computers & fluids

DOI:
[10.1016/j.compfluid.2016.06.026](https://doi.org/10.1016/j.compfluid.2016.06.026)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2016

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Cifani, P., Michalek, W. R., Priems, G. J. M., Kuerten, J. G. M., van der Geld, C. W. M., & Geurts, B. J. (2016). A comparison between the surface compression method and an interface reconstruction method for the VOF approach. *Computers & fluids*, 136, 421-435. <https://doi.org/10.1016/j.compfluid.2016.06.026>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

A comparison between the surface compression method and an interface reconstruction method for the VOF approach

P. Cifani^{a,*}, W.R. Michalek^b, G.J.M. Priems^b, J.G.M. Kuerten^{b,a}, C.W.M. van der Geld^b, B.J. Geurts^{a,b}

^a Multiscale Modeling and Simulation, Faculty EEMCS, University of Twente, P.O. Box 217, 7500AE Enschede, The Netherlands

^b Eindhoven University of Technology, P.O. Box 513, 5600MB Eindhoven, The Netherlands

ARTICLE INFO

Article history:

Received 1 March 2016

Revised 19 June 2016

Accepted 30 June 2016

Available online 1 July 2016

Keywords:

Multiphase flow

VOF

PLIC

Interface compression

OpenFOAM®

ABSTRACT

In direct numerical simulations of multiphase flows, based on the Volume of Fluid (VOF) approach, the advection of the volume fraction field is a crucial point. The choice of the discretisation scheme for the transport of the volume fraction is decisive for an accurate description of surface dynamics. In this paper we assess two numerical methods: a high order discretisation scheme, namely the surface compression scheme, and an interface reconstruction scheme based on a piecewise linear interface calculation (PLIC). We compare accuracy, convergence rate and computational cost of these methods with results from literature. The comparative study includes reference 2D and 3D advection test cases. Moreover, the advection algorithm is tested coupled to an incompressible Navier–Stokes solver and used to simulate a rising bubble in a liquid for different Eötvös and Reynolds numbers. We establish via the advection tests and through the study of rising bubbles that the PLIC method converges to second order while the compression method fails to converge systematically. The computational overhead of both methods is negligible compared to an incompressible flow solver to which it might be coupled.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

The present study is concerned with isothermal, incompressible two-phase gas–liquid flows. The interface between the two phases is represented implicitly by adopting the Volume-of-Fluid (VOF) method [1] which has been extensively used in the CFD community [2–4] and has been shown to be robust and able to capture topologically complex interfaces. Here we present a careful comparison between two numerical methods, representative of two classes of numerical methods for the advection of the interface between two phases: a VOF–PLIC algorithm with split advection technique [5] and the scheme presented by [6] based on interface compression. It is important to test the advection algorithms for different physical parameters of the fluid flow in order to assess the reliability of numerical predictions. Establishing the accuracy of the numerical method for the advection of the phase indicator function in real-life problems, such as a rising bubble in a viscous liquid, is not straightforward since no analytical solution is available. This paper addresses the issue of selecting the best numerical prediction method from two classes of methods, not only by assessing

the accuracy of both methods for numerical benchmark cases but also by showing the viability of the method for complex flows with a deforming interface. The benefit of this combined comparison is to establish the numerical features of the two schemes on a simplified set up where exact solutions are known and at the same time to assess the validity of the numerical results on problems of physical relevance.

The shape regimes for bubbles and drops in gravitational motion through a liquid can be determined as functions of buoyancy, surface tension and viscosity expressed by the Reynolds (Re) and the Eötvös (Eo) numbers. For example, a single buoyant bubble at sufficiently low Eo , remains close to a sphere. For high Eo value buoyancy is dominant over surface tension, causing the bubble to deform, creating wakes that lead to a different flow dynamics. Numerous numerical studies are available in literature on rising bubbles under different flow conditions [7–13]. Here, the numerical simulations are carried out in the open-source code OpenFOAM® [14] where the compression method is present as a built-in feature, while the VOF–PLIC algorithm has been implemented by the authors. The data from the benchmark reported in [15] on a rising bubble in a viscous liquid, are used as point of reference.

A crucial point in any numerical method for direct numerical simulation of multiphase flow is the advection of the phase indicator function. In the VOF approach, the discrete indicator func-

* Corresponding author.

E-mail address: p.cifani@utwente.nl (P. Cifani).

tion is the volume fraction occupied by a particular fluid within one cell. This function is not smoothly distributed across the interface where we have a sharp discontinuous field between 0 and 1. Conventional convective differencing schemes can either lead to excessive smearing of the interface thickness or to violation of the boundedness of the volume fraction to $[0, 1]$. Therefore, the choice of the discretisation scheme for the transport of the phase indicator has a large influence on the surface dynamics and a careful assessment is required. A comprehensive set of assessment steps is introduced in this paper and applied to the compression method and to PLIC.

Another important aspect of the numerics is the mapping of the pressure drop onto the underlying computational grid [16,17], and the discretisation of the surface tension, which requires the computation of the curvature. A poor estimate of the curvature often leads to unphysical velocities around the interface, resulting in a substantial loss of accuracy. Several techniques have been developed [4,18,19]. In this paper, we focus on the transport of the phase indicator function since an accurate advection algorithm delivers an accurate representation of the interface and the geometrical properties related to it. These geometrical properties, in turn, are key for pressure/surface tension mappings.

For the numerous techniques that have been developed to treat this problem, two classes can be distinguished: methods that discretise the hyperbolic equations for the advection of a step function by using high resolution and tailored discretisation schemes [20–25]; and methods based on geometric interface reconstruction as well as geometric evaluation of fluxes [17,26,27].

High resolution differencing techniques are usually based on a combination of higher and lower order discretisation schemes of the convective term. The discontinuity of the volume fraction at the interface can lead to numerical oscillations. To prevent this, non-linear differencing schemes are employed to achieve a discrete operator of positive type and at the same time an accuracy higher than first order. These methods have no limitations for grid topology and can be mass preserving. The drawback is a loss of accuracy due to inevitable smearing out of the interface during the advection process due to numerical diffusion at the interface.

Geometric advection schemes are characterised by an accurate representation of the interface by a number of curves/surfaces, to match the local volume fraction. The reconstructed interface is subsequently advected. The interface reconstruction method proceeds in a number of steps. A major effort is related to the estimation of the intersection of the interface with a grid cell. This procedure is commonly combined with a split advection algorithm [2,28] in which the transport of the volume fraction field is sequentially carried out along each orthogonal direction separately. While our method uses the split advection algorithm along Cartesian coordinates, making it suitable for orthogonal meshes, unsplit advection algorithms can be developed that are suitable also on unstructured meshes [29]. A review of split advection algorithms is beyond the purpose of this work and can be found in [30].

An important high-order method is the interface compression method presented by Rusche [6] and also adopted in [13], while the piece wise linear interface calculation (PLIC) algorithm with a split advection technique [5,31] is representative of the class of geometric methods. The present paper presents a comparison between these two methods as these are representative of the two classes, making this study meaningful also for other methods. On the one hand the compression method is less accurate but also less expensive. The platform we work with is OpenFOAM®, which includes the multiphase solver interFoam based on VOF in which the compression method is used for the advection of the volume fraction. For this study, a new solver for both 2D and 3D geometrical VOF algorithms was developed within the OpenFOAM® framework.

The implementation is done in the C++ programming language for high modularity and reusability.

The organisation of this paper is as follows. In Sections 2 and 3 we present the mathematical formulation of the problem and the resulting numerical algorithm, respectively. For the VOF–PLIC reconstruction procedure, we adopt the method proposed in [32]. This procedure makes use of analytic relations for the computation of intersections between an interface and a computational cell, which allows a robust implementation. The splitting algorithm is a sequence of implicit and explicit Eulerian time-stepping calculations of the fluxes as described in [5]. In Section 4, the methods are validated for several so-called advection test cases where a velocity field is imposed externally and only the transport of the volume fraction is solved. Comparisons with literature and with the results obtained from the CFD tool ANSYS Fluent [33] are made. Subsequently, in Section 5, a simulation of a rising bubble in a viscous liquid is carried out. The parameters of the simulation are as given in the benchmark [15], in which the simulation has been performed with three independent codes. The accuracy and the convergence of the two advection algorithms is investigated by studying the time evolution of the circularity and the bubble shape. The results are then compared with the ones in the benchmark. Concluding remarks are collected in Section 6.

2. Mathematical formulation

In this paper multiphase flow is simulated by solving a single set of equations for the whole domain. The interaction between the two phases is incorporated in the equations as concentrated source terms acting only at the interfaces. The Navier–Stokes and the continuity equations for the motion of a viscous, incompressible, immiscible two-fluid system can be written as [34]

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) = -\nabla p + \rho \mathbf{g} + \nabla \cdot (2\mu \mathbf{D}) + \sigma k n \delta(n) \quad (1)$$

$$\nabla \cdot \mathbf{u} = 0 \quad (2)$$

Here $\mathbf{u} = (u, v, w)$ is the velocity field, ρ is the mass density of the fluid, μ is the dynamic viscosity, $\mathbf{D}$ is the deformation tensor, $D_{i,j} = (\partial_i u_j + \partial_j u_i)/2$, σ is the surface tension coefficient, k is the curvature of the interface and $\mathbf{n}$ is the unit normal to the interface, $\mathbf{g}$ is the gravitational acceleration. The Dirac delta function $\delta(n)$, with n being the local coordinate normal to the interface in the direction of $\mathbf{n}$, locates the surface term at the interface only. The density and the viscosity are assumed to be constant in each phase separately. The spatial distribution of the phases is defined by the indicator function $f(\mathbf{x}, t)$. The function f is a Heaviside function which is equal to 1 if $(\mathbf{x}, t)$ is in one of the phases and equal to 0 in the other. At a numerical level f can have a smooth transition between these values and in the VOF approach it represents the volume fraction of one phase. Mass density and dynamic viscosity are expressed by

$$\rho(\mathbf{x}, t) = f\rho_1 + (1 - f)\rho_2 \quad (3)$$

$$\mu(\mathbf{x}, t) = f\mu_1 + (1 - f)\mu_2 \quad (4)$$

where ρ_1 , ρ_2 and μ_1 , μ_2 are the mass density and the dynamic viscosity of the two separate fluids, respectively. In the absence of phase change, the phase indicator simply follows the fluid flow. The function f is then constant along particle paths, which implies that the material derivative is zero, leading to the following equation:

$$\frac{\partial f}{\partial t} + \mathbf{u} \cdot \nabla f = 0. \quad (5)$$

Fig. 1. Schematic 1D stencil where the dashed line defines the boundary of a control volume with the grid point P as its center. The points W and E are the centroids of the neighbouring cells nb to the West and to the East. Corresponding cell faces are labeled w and e .

The set of Eqs. (1), (2) and (5) describes the multiphase flow. The next section focuses on the numerical discretisation of the governing equations.

3. Numerical algorithm

In this section we first describe the discretisation and the numerical solution of the Navier–Stokes equations. Subsequently, we turn the attention to the discretisation of the surface tension term after which follows a detailed description of the numerical algorithms for the advection of the indicator function (5).

3.1. Navier–Stokes algorithm

The Navier–Stokes equations are cast in a momentum-conservative formulation on a collocated arrangement of the grid. The primary variables, i.e., $\mathbf{u}$, p and f , are then stored in cell centres. The indicator function is represented by the value of the volume fraction f in each computational cell. The advection Eq. (5) is solved first using an explicit time integration method where the velocity field is computed at time $t^n = n\Delta t$ (see Sections 3.3 and 3.4), with Δt the time-step. Once the volume fraction f^{n+1} is known, the fluid properties are updated using (3) and (4). A second order backward differencing scheme in time (BDF2) of (1) combined with imposing (2) at the new time level leads to

$$\frac{1}{\Delta t} \left[\frac{3}{2} (\rho \mathbf{u})^{n+1} - 2 (\rho \mathbf{u})^n + \frac{1}{2} (\rho \mathbf{u})^{n-1} \right] = \mathbf{R}^{n+1} \quad (6)$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0$$

where we take $\mathbf{R}^{n+1} = -\nabla p^{n+1} - \nabla \cdot (\rho \mathbf{u} \mathbf{u})^{n+1} + \nabla \cdot (2\mu \mathbf{D})^{n+1} + (\sigma \kappa \delta(n))^{n+1} - (\rho \mathbf{g})^{n+1}$. A semi-discrete form of the momentum equation, using the finite volume method, reads

$$a_p \mathbf{u}_p = H(\mathbf{u}) - \nabla \hat{p} + F. \quad (7)$$

where $\mathbf{u}_p$ is the discrete velocity vector in the centre of cell P . The term on the left-hand side and the second and the third term on the right-hand side of Eq. (7) are computed at time $n+1$. The vector $H(\mathbf{u})$ is defined as $H(\mathbf{u}) = \sum_{nb} a_{nb} \mathbf{u}_{nb}^{n+1} + \frac{1}{\Delta t} [2(\rho \mathbf{u})_p^n - \frac{1}{2}(\rho \mathbf{u})_p^{n-1}]$ in which the sum runs over all nearest neighboring cells of P . The coefficients a_p and a_{nb} result from the discretisation of the time derivative, the diffusion term $\nabla \cdot (2\mu \mathbf{D})^{n+1}$ and the convective term $\nabla \cdot (\rho \mathbf{u} \mathbf{u})^{n+1}$ for all the neighbouring cells nb of a generic cell P . Fig. 1 shows a schematic example of a 1D grid.

A full description of the coefficients a_p and a_{nb} can be found in [6]. We made use of the modified pressure $\hat{p} = p - \rho \mathbf{g} \cdot \mathbf{x}$, with $\mathbf{x}$ the position vector. This form allows an efficient numerical treatment of the steep density jump at the interface, on a collocated grid, as shown in [6]. The vector F contains the discretisation of the remainder of the gravity term $-\mathbf{g} \cdot \mathbf{x} \nabla \rho$ and the surface tension term. The treatment of the source term F will be explored in more detail in Section 3.2. From (7) a discrete velocity field at cell faces can be obtained as follows [35]

$$\mathbf{u}_f = \left(\frac{H(\mathbf{u})}{a_p} \right)_f - \left(\frac{1}{a_p} \right)_f [\nabla_f \hat{p} - (F)_f] \quad (8)$$

where $()_f$ indicates quantities interpolated at cell faces and ∇_f is a standard face-centred gradient operator. The application of the

continuity constraint (2) to (8) leads to the Poisson equation for the pressure:

$$\nabla \cdot \left[\left(\frac{1}{a_p} \right)_f \nabla_f \hat{p} \right] = \nabla \cdot \left(\frac{H(\mathbf{u})}{a_p} \right)_f + \nabla \cdot \left[\left(\frac{1}{a_p} \right)_f (F)_f \right] \quad (9)$$

From the solution of the pressure field we can build a discrete divergence-free velocity by using interpolation (8). This step, analogous to the explicit correction introduced by Rhie and Chow [36], avoids the classical problem of decoupling of the pressure and velocity field on a collocated grid [35].

The PISO algorithm [37], is used to couple the system of Eqs. (7) and (9). A first predictor step is employed to obtain an intermediate velocity field by solving Eq. (7) where the pressure is the one at the previous time-level. This step requires the solution of a linear system. Subsequently, a number of explicit corrector steps are performed. The continuity constraint is enforced through (9) after which follows a new velocity field that is explicitly computed from (7). The number of explicit corrector steps influences the accuracy of the solution and at least two steps are necessary to match the second order accuracy of the time integration scheme [37]. This algorithm has been shown to be particularly suitable for implicitly discretised Navier–Stokes equations.

A centred finite-difference operator is used for all the spatial derivatives. The explicit time integration (5) implies the time step restriction [38]:

$$\Delta t \leq \min_D \left(c \frac{h}{|u|} \right) \quad (10)$$

where h is the grid size, D is the spatial domain, and c a coefficient that depends on the dimensions of the problem and on the particular discretisation used for the transport of the volume fraction field. According to Brackbill [18], the time step has to resolve the capillary wave, which leads to the restriction $\Delta t < \sqrt{(h^3(\rho_1 + \rho_2)/(4\pi\sigma))}$. This condition is deduced in the ideal situation of an interface separating two inviscid fluids with zero mean flow and zero gravity. For the simulations analysed in this study involving of a rising bubble in a viscous liquid, we did not find any unstable behaviour for time steps exceeding this stability restriction. Stable results are also found in the benchmark [15] under the same time-step conditions.

3.2. Discretisation of surface tension

The surface tension term is modelled with the continuum-surface-force (CSF) approach proposed by Brackbill [18]. The delta function $\delta(n)\mathbf{n}$ is replaced by the gradient of a smoothed volume fraction field $\tilde{f}$:

$$\sigma k \delta(n)\mathbf{n} \approx \sigma k \nabla \tilde{f} \quad (11)$$

where the curvature k is computed as

$$k \approx \nabla \cdot \hat{\mathbf{n}} \quad (12)$$

with

$$\hat{\mathbf{n}} = \frac{\nabla \tilde{f}}{|\nabla \tilde{f}|} \quad (13)$$

This approach is known to suffer from *spurious* currents for the case of a static droplet under zero gravity. As pointed out in [16,39–41] the source of this parasitic velocity can be traced back to two main causes: an inaccurate computation of the curvature and a discrete imbalance between pressure gradient and surface tension. Under zero flow and gravity conditions equilibrium reads $\nabla p = \sigma k \delta(n)\mathbf{n}$. At a numerical level this condition can be violated while estimating the surface tension at the face centres in Eq. (9) by simply averaging the values from the cell centres. In

OpenFOAM® a discrete equilibrium between surface tension and pressure gradient is obtained by the following procedure. In the Poisson Eq. (9) the contribution from the surface tension is computed by applying the operator ∇_f to the volume fraction field. The same relation is used for the face-centred velocity (8), so that $(F)_f$ is replaced by

$$(F)_f = \sigma k_f \nabla_f \tilde{f} \quad (14)$$

Differently, the cell-centred velocity, obtained from (7), is computed by replacing the term of the pressure gradient and the source term F in the following way:

$$-\nabla \hat{p} + F = \left| -\nabla_f \hat{p} + \sigma k_f \nabla_f \tilde{f} \right|^{f \rightarrow c} \quad (15)$$

where $| \cdot |^{f \rightarrow c}$ represents the average of the faces of a computational cell to its central node. This procedure makes the discrete gradient of the pressure and volume fraction compatible and it has been shown to significantly reduce the spurious currents [4,16]. The same steps are applied to the density gradient in the gravity term.

As Eqs. (12) and (13) show, an accurate computation of the curvature and the normal vector to the interface relies on an accurate numerical discretisation of the first and second derivative of $\tilde{f}$. The steep gradients of the volume fraction at the interface, that characterise the transition from one phase to the other, can lead to an overall loss of accuracy of the numerical solution. This point does not seem to be addressed in OpenFOAM®, where no regularisation of the delta function in (11) is applied. In this study, a smoothing technique based on the convolution of the volume fraction field f with a kernel Φ is employed [42]. In particular, consider the following problem:

$$\begin{aligned} \frac{\partial \varphi(x, \tau)}{\partial \tau} &= k \nabla^2 \varphi(x, \tau) \quad \text{in } \mathbb{R}^n \times [0, \infty) \\ \varphi(x, 0) &= g(x) \quad \text{in } \mathbb{R}^n \end{aligned} \quad (16)$$

with $g(x)$ continuous and bounded and $\lim_{|x| \rightarrow \infty} \varphi(x, \tau) = 0$. The solution to problem (16) is the spatial convolution

$$\varphi(x, \tau) = g * \Phi = \int_{\mathbb{R}^n} g(s) \Phi(x - s, \tau) d^n s \quad (17)$$

where the Green's function $\Phi(x, \tau)$ has the following expression

$$\Phi(x, \tau) = \frac{1}{(4\pi k \tau)^{n/2}} e^{-\frac{|x|^2}{4k\tau}} \quad (18)$$

with $|x|^2 = \sum_i^n (x_i)^2$. If we set g equal to the volume fraction f at the current time level, we can obtain a spatially filtered field $\tilde{f}$ by solving the unsteady diffusion problem (16). The fictitious time τ and diffusivity k control the smoothing length. An explicit time integration of (16) has the advantage of having a low computational cost per grid-point and time step and can easily be implemented. The parameter τ is determined such that at a distance equal to the smoothing length s_l the value of the kernel is close to zero. This is obtained by

$$\frac{\Phi(s_l, \tau)}{\Phi(0, \tau)} = \epsilon \quad (19)$$

where ϵ is an arbitrary small number, that we choose equal to 10^{-3} . Eq. (19) gives

$$\tau = \frac{s_l^2}{12k \ln(10)} \quad (20)$$

In practice, the volume fraction needs to be spread over a few computational cells only, since the CFS approach requires a transition region of f from 1 to 0 on a narrow band across the interface. A small number of a time-steps is then needed. The space discretisation error is therefore dominant, so that an Euler forward scheme can efficiently be used for the time marching of (16), while

Fig. 2. Schematic of 2D geometrical flux. The shaded region is occupied by one phase inside the cell i, j . The portion of this region inside the dashed lines will be convected by the velocity $u^{i+1/2, j}$ to the cell $i+1, j$.

a central differencing scheme is used for the discretisation of the Laplacian.

A smoothing length of about 3 cells is used for the evaluation of a centred 3 point stencil approximation of the first derivative of $\tilde{f}$ in all the advection cases analysed in Section 4. Preliminary tests have shown that values much larger than 3 would result in a loss of accuracy due to excessive smearing of sharp edges. Values smaller than 3 would lead to computing derivatives on a too steep function resulting in a large discretisation error. The computation of the curvature, via Eq. (12), is much more sensitive than the computation of the normal $\mathbf{n}$ to the variations of the smoothing length. For this reason, a systematic study of the influence of s_l is conducted in Section 5.1 on a simulation of rising bubble, where k appears in the surface tension term. The filtering of the volume fraction field f was found to be crucial for the convergence of the numerical solution, as observed in [43].

3.3. VOF advection algorithm with piecewise-linear geometric method

The transport Eq. (5) is solved by using a piecewise-linear geometric VOF scheme [27]. Geometrical Volume-of-Fluid methods consist of two steps: the interface between the two fluids is first approximated according to a defined shape; next, based on the approximate interface, the fluxes are evaluated and the interface is advected.

In the piecewise-linear VOF scheme the interface in a grid cell is defined by

$$\mathbf{n} \times \mathbf{x} = q \quad (21)$$

where $\mathbf{n}$ is the normal vector to the interface, that is computed from (13), $\mathbf{x}$ is the position vector and q a parameter related to the intersection points of the interface with the coordinate axes. Eq. (21) defines a line/plane, respectively in 2D and 3D. The volume fraction in computational cells that satisfy $0 < f < 1$ is equal to the area/volume on the side of the line/plane of the reconstructed interface (shaded region in Fig. 2) divided by the cell area/volume, according to the relation

$$f = \mathcal{T}(\mathbf{n}, q) \quad (22)$$

Vice-versa, given the field f and the normal vector $\mathbf{n}$, the parameter q defines a unique line/plane that satisfies the relation

$$q = \mathcal{T}^{-1}(\mathbf{n}, f) \quad (23)$$

We remark that f in (22) and (23) is the sharp volume fraction field since the use of its smoothed version $\tilde{f}$ would lead to smearing of the interface, violating so an essential feature of this method. The functions $\mathcal{T}$ and $\mathcal{T}^{-1}$ have been implemented in OpenFOAM®, and they correspond to the routines developed by

Scardovelli [32]. These are based on analytical root finding formulas for quadratic/cubic equations and result in a robust numerical implementation. A detailed description is given in the Appendix.

Once the interface is reconstructed by (23), the geometrical fluxes are computed. A sketch of the flux estimation is illustrated in Fig. 2. The grey area is the volume fraction of one phase in the computational cell (i, j) . The shaded region inside the dashed line will be convected by the face velocity $u_{i+1/2,j}$ to the neighbour cell $(i+1, j)$. The volume of this region is an estimate of the flux $(uf)_{i+1/2,j}$, which can be computed using relation (22).

The time-marching of the transport Eq. (5) is carried out by using an operator split method, where the advection of the volume fraction is done sequentially along each coordinate direction. In this study, we make use of the algorithm proposed by Puckett [5]. A 2D time discretisation of (5) reads

$$\begin{aligned} f^* &= f^n - \Delta t \frac{\partial(uf^n)}{\partial x} + \Delta t f^* \frac{\partial u}{\partial x} \\ f^{n+1} &= f^* - \Delta t \frac{\partial(vf^*)}{\partial y} + \Delta t f^* \frac{\partial v}{\partial y} \end{aligned} \quad (24)$$

where f^* is the volume fraction at an intermediate time level. Since the relations (22) and (23) are only valid for values of the volume fraction bounded between 0 and 1 the additional term $f^*(\partial_x u + \partial_y v)$ is added to limit the possible undershoots or overshoots that would occur after a single direction sweep. The correction term vanishes in the total change $(f^{n+1} - f^n)/\Delta t$ provided the velocity is divergence free. The corresponding 3D algorithm reads [31]

$$\begin{aligned} f^* &= f^n - \Delta t \frac{\partial(uf^n)}{\partial x} + \Delta t f^* \frac{\partial u}{\partial x} \\ f^{**} &= f^* - \Delta t \frac{\partial(vf^*)}{\partial y} + \Delta t f^{**} \frac{\partial v}{\partial y} \\ f^{***} &= f^{**} - \Delta t \frac{\partial(wf^{**})}{\partial z} + \Delta t f^{***} \frac{\partial w}{\partial z} \\ f^{n+1} &= f^{***} - \Delta t \left(f^* \frac{\partial u}{\partial x} + f^{**} \frac{\partial v}{\partial y} + f^{***} \frac{\partial w}{\partial z} \right) \end{aligned} \quad (25)$$

Similar to the two-dimensional case, f^* , f^{**} , f^{***} are volume fraction fields after each advection sweep and the second term in the last expression of (25) will subtract the contribution of the correction terms at the final time level $n+1$. Furthermore, in order to minimise asymmetries of the splitting technique, the order of the advection steps along each coordinate direction is alternated. In particular, all possible permutations are performed. There are 2 of these permutations for two-dimensional problems and 6 for three-dimensional problems.

This VOF advection scheme has been shown to be not strictly mass conserving [2], since small overshoots or undershoots can still appear during the advection steps. In practice, the error in the mass conservation has been found to be small compared to the other sources of error [4]. A quantitative analysis of the numerical errors of the advection algorithm is made in Section 4.

3.4. VOF advection algorithm with interface compression method

In conservative form VOF Eq. (5) reads

$$\frac{\partial f_i}{\partial t} + \nabla \cdot (f_i \mathbf{u}_i) = 0, \quad i = 1, 2 \quad (26)$$

In this subsection, we derive the surface compression method, and distinguish between the velocity fields $\mathbf{u}_1$ and $\mathbf{u}_2$ and the volume fractions f_1 and f_2 of the two fluids. It is sufficient to consider only the evolution in time of one of the two volume fractions, e.g., f_1 advected by velocity $\mathbf{u}_1$. If the interface is maintained sharp, as in the method described in the previous section, the velocity $\mathbf{u}_1$ is assumed to be equal to the total velocity $\mathbf{u}$. This is not true

in the surface compression approach, where the interface is always smeared out over a few computational cells. According to [6,44] the total velocity $\mathbf{u}$ is defined as

$$\mathbf{u} = f_1 \mathbf{u}_1 + f_2 \mathbf{u}_2 \quad (27)$$

and the relative velocity $\mathbf{u}_r$ as

$$\mathbf{u}_r = \mathbf{u}_1 - \mathbf{u}_2. \quad (28)$$

Adding Eq. (27) multiplied by f_1 to Eq. (28) multiplied by $f_1(1 - f_1)$ we have

$$\mathbf{u}_1 f_1 = \mathbf{u} f_1 + f_1(1 - f_1) \mathbf{u}_r. \quad (29)$$

Inserting (29) in Eq. (26) written for fluid 1 yields

$$\frac{\partial f_1}{\partial t} + \nabla \cdot (f_1 \mathbf{u}) + \nabla \cdot [\mathbf{u}_r f_1(1 - f_1)] = 0 \quad (30)$$

The term $\nabla \cdot [\mathbf{u}_r f_1(1 - f_1)]$ is the surface compression term which is only active in the vicinity of the interface because of the product $f_1(1 - f_1)$. The additional term in Eq. (30) contributes to a higher interface resolution which partially prevents the smearing out of the interface. The relative velocity $\mathbf{u}_r$ is not directly available from the one-fluid formulation because we solve for the whole velocity $\mathbf{u}$. The numerical implementation of the relative used in OpenFOAM® is given by

$$(\mathbf{u}_r)_f = \mathbf{n}_f \min \left(c_\gamma \left| \frac{\phi}{|\mathbf{S}|} \right|, \left| \frac{\phi}{|\mathbf{S}|} \right|_{\max} \right) \quad (31)$$

where the magnitude of $(\mathbf{u}_r)_f$ is bounded to the maximum face velocity in the flow field and its direction is aligned with the normal to the interface $\mathbf{n}_f$ defined as [44]

$$\mathbf{n}_f = \frac{(\nabla f)_f}{|(\nabla f)_f + \delta|} \quad (32)$$

with δ a stabilisation factor. In (31) $\mathbf{S}$ is the cell face oriented area, ϕ is the the flux $\mathbf{u}_f \cdot \mathbf{S}$ and c_γ is a tunable coefficient that determines the effect of the compression term. The model (31) relies on the definition of relative velocity (28), so that when the velocity $\mathbf{u}_2$ at the interface is small the relative velocity will be close to $\mathbf{u}_1$. If the velocities are of the same order of magnitude the coefficients c_γ controls the intensity of the additional term in (30). We have found that for values of c_γ considerably lower than 1 the interface is excessively smeared out. On the other hand, increasing c_γ above 1 the resulting interface is distorted and numerical instability may arise. Therefore, for all simulations we have chosen $c_\gamma = 1$. The same trends were observed in [13], and a value of 1 is also recommended by the OpenFOAM® user guide. The Van Leer scheme is used for the discretisation of the convective term $\nabla \cdot (\mathbf{u} f_1)$, while for the nonlinear convective term $\nabla \cdot (\mathbf{u}_r f_1(1 - f_1))$ a quadratic flux limiter is used [14].

Eq. (30) is solved explicitly by a multidimensional universal limiter solver (MULES) [14]. This solver is designed to preserve the boundedness of the volume fraction independently of the mesh topology and the numerical scheme of the convective terms.

In this section we have presented the numerical algorithm detailing the two main cores of it: the Navier–Stokes solver and the advection of the volume fraction equation. In the next section a set of numerical solution of Eq. (5) is generated and analysed in order to assess the accuracy and the convergence of the two methods presented here.

4. Pure advection test cases

In this section we test the numerical algorithms described in Section 3. A set of advection test cases is performed in order to compare the interface compression method with the piecewise-linear geometric method. The advection test cases are kinematic

simulations where Eq. (5) is solved, subject to an assumed velocity field. This allows to isolate the numerical error of the advection algorithm from that due to the computation of the velocity field itself. The latter, fully coupled, situation will be considered in Section 5.

In all cases we present in this section, the external velocity field is defined such that the body that is being advected returns to its initial shape after some time. This gives a straightforward measure of the accumulated error in a simulation. To be consistent with literature we use the following error measures:

$$E_1 = \sum_i |f_i - \bar{f}_i| h^n \quad (33)$$

$$E_2 = \frac{\sum_i |f_i - \bar{f}_i|}{\sum_i \bar{f}_i} \quad (34)$$

$$E_3 = \frac{\sum_i (f_i - \bar{f}_i)}{\sum_i \bar{f}_i} \quad (35)$$

where the summation is over all grid points, h is the constant grid size and n is equal to 2 or 3 for 2D and 3D simulations, respectively. In these error measures f is the numerical volume fraction at the end of the simulation and $\bar{f}$ is the volume fraction imposed as initial condition of the simulation. The errors (33) and (34) are used to estimate the truncation error of the numerical method while (35) is a measure for the mass conservation. The rate of convergence with grid refinement is indicated in all subsequent tables in italics and is computed by

$$R = \frac{\log((E_k)_h / (E_k)_{h/2})}{\log(2)} \quad (36)$$

where $k = 1, 2, 3$ and $(E_k)_h$ is an error measure on a grid size h . The convergence rate R is equal to 1 for a first order method and 2 for a second order method. For brevity, hereafter we refer to the interface compression method as *interFoam* and to the geometrical VOF as *gVoFoam*.

For both methods, the advection algorithm requires the computation of the vector normal to the interface as shown in (31), (22) and (23). In the case of *interFoam*, this is achieved by (32). In the case of *gVoFoam*, $\mathbf{n}$ is computed from the smoothed volume fraction field by (13).

4.1. Rotation of a 2D slotted disk

A solid body rotation of a slotted disk, often referred to as Zalesak's test [20], is simulated. The set up we employed here corresponds to the one used in [24]. The circle has a radius of 0.5 and it is placed inside a square domain of length 4 centred at (2, 2.75), with 200 cells along each dimension. The width and the length of the slot are, respectively, equal to 0.12 and 0.6. The disk completes a full rotation in 2524 time-steps, with a maximum Courant number in the domain of about 0.5. The velocity field is the following:

$$u = -\omega(y - y_c)$$

$$v = \omega(x - x_c)$$

where ω is the angular velocity and (x_c, y_c) the centre of the square.

In Fig. 3 the final numerical shape is shown together with the exact solution. The result obtained with *gVoFoam* is visibly more accurate than that obtained with *interFoam*. A quantitative analysis in Table 1, shows that the results of *gVoFoam* are comparable with literature, while with *interFoam* we observe a value of E_2 about 5 times larger.

Fig. 3. Final shape in the Zalesak test: the black line is the exact solution; the red line is the result from *interFoam*; the blue line is the result from *gVoFoam*. In this simulation a resolution of 200×200 was used with a maximum CFL number equal to 0.5. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
Zalesak's test [20]; grid: 200×200 .

VOF algorithms	Error (34)	Error (33)	Error (35)
Youngs [24]	1.09×10^{-2}	/	/
Elvira/Lagrangian [30]	1.00×10^{-2}	/	/
<i>gVoFoam</i>	1.36×10^{-2}	9.71×10^{-3}	-7.76×10^{-14}
<i>interFoam</i>	6.61×10^{-2}	4.71×10^{-2}	-7.66×10^{-16}

The error in the conservation of mass E_3 is equal to -7.76×10^{-14} and 7.66×10^{-16} respectively for *gVoFoam* and for *interFoam*. The latter is mass conservative by construction, and this result establishes confidence that its implementation in OpenFOAM is reliable. The *gVoFoam* method is known to suffer from small undershoots/overshoots of the volume fraction [2]. On the contrary, in this particular test case, the velocity field satisfies $\partial u / \partial x = \partial v / \partial y = 0$. Therefore the 1D divergence correction terms in (24) are zero. This makes the method mass conserving in this particular case, which is well realized as E_3 values indicate.

4.2. Deformation field - 2D

A circle of radius 0.15 inside a unit sized box, with centre initially at point (0.5, 0.75), is subjected to the following velocity field:

$$u = \sin(2\pi y) \sin^2(\pi x) \cos\left(\frac{\pi t}{T}\right)$$

$$v = -\sin(2\pi x) \sin^2(\pi y) \cos\left(\frac{\pi t}{T}\right)$$

This test case has been used by many researchers [2,24,30,45]. With respect to Zalesak's test, the presence of shear leads to deformation of the body of fluid. The cosine term, depending on time, reverses the velocity field at $t = T/2$ so that the deformed circle should return back to its initial shape at $t = T$. The CFL number, based on the maximum velocity magnitude, is chosen equal to 1 to be consistent with literature. Table 2 shows the values of the error (33) at $T = 8$ for different grid refinement. The results are compared with [2]. The errors obtained with *gVoFoam* are on the same order of magnitude as reported in [2] and second order convergence is observed. An unsplit time integration scheme and computation of the interface normal based on a minimisation of error [46] is used in [2]. That explains the small differences observed in Table 2.

Table 2

Error (33) in the 2D deformation field ($CFL = 1$). The convergence rate R is reported in italic.

Grid	$T = 8$	
	gVoFoam	Rider and Kothe [2]
32^2	5.90×10^{-2} 2.03	4.78×10^{-2} 2.78
64^2	1.44×10^{-2} 2.37	6.96×10^{-3} 2.27
128^2	2.78×10^{-3}	1.44×10^{-3}

Fig. 4. Contour level $f = 0.5$ for the 2D deformation field test on a 128×128 grid: the black line is the exact solution; the red line is the result from interFoam; the blue line is the result from gVoFoam. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 3

Error (33) and (35) in the 2D deformation field ($CFL = 0.5$). The convergence rate R is reported in italic. As final time we adopt $T = 8$.

Grid	E_1		E_3	
	gVoFoam	interFoam	gVoFoam	interFoam
32^2	8.10×10^{-2} 2.16	1.04×10^{-1} 0.22	1.32×10^{-3} 3.15	2.06×10^{-14}
64^2	1.80×10^{-2} 2.61	8.92×10^{-2} 0.67	1.49×10^{-4} 4.21	3.59×10^{-14}
128^2	2.95×10^{-3}	5.58×10^{-2}	8.04×10^{-6}	6.12×10^{-14}

A CFL number close to 1 results in an unstable solution for the time integration scheme used in interFoam [6]. To test the interface compression method also away from this instability, we also considered $CFL = 0.5$ and performed a grid refinement study. Fig. 4 shows the final shape of the circle on a uniform grid with 128×128 cells while Fig. 5 shows the deformed shape at $T/2$. Clearly at the final time $T = 8$, interFoam fails to reproduce the exact solution and is qualitatively in error, while the final result obtained by gVoFoam is very close to a circle. At $t = T/2$, where the deformation is maximum, a highly fragmented interface is observed using the interface compression method while gVoFoam appears more reliable. A quantitative comparison is reported in Table 3. We notice that interFoam is superior in terms of conservation of mass which is not remarkable as this aspect is built into the method. We also notice that interFoam does not show convergence in E_1 with grid refinement suggesting even a systematic error. The error in the mass conservation with the geometrical VOF is small for the coarsest grid considered and decreases about one order of magnitude when halving the mesh size.

Table 4

Error (33) in the 3D deformation field. The convergence rate R is reported in italic.

grid	gVoFoam	Liovic et al. [45] (Youngs + direction-split)
32^2	$9.18e-03$ 1.56	$7.71e-3$ 1.47
64^2	$3.11e-03$ 1.78	$2.78e-3$ 1.87
128^2	$9.06e-04$	$7.58e-4$

As a further validation of the developed gVoFoam implementation, we compare PLIC results against the commercial CFD tool ANSYS Fluent [33] for the case $T = 8$ and $CFL = 0.5$. Fig. 6 shows the error E_1 as a function of the grid size h . The results of both methods are within 30% of each other with somewhat better convergence of gVoFoam at higher resolutions. This independent test provides an additional confirmation of the reliability of the gVoFoam implementation, while at the same time establishing that also the Fluent PLIC implementation in 2D performs as expected for this test case.

Fig. 7 (a) shows the computational time as a function of the resolution on one core of a 2.2 GHz Intel Xeon Processor. gVoFoam is about 3 times slower than interFoam on the finest grid. But if we look at the computational time as a function of the error E_1 (Fig. 7(b)), we see that for the same grid gVoFoam is about two orders of magnitude more accurate than interFoam.

4.3. Deformation field - 3D

In order to validate the geometrical VOF method in three dimension, we test the solver gVoFoam on the case proposed by [47], and later used by [45,48–50]. A sphere of radius 0.15 is centred at (0.35, 0.35, 0.35) in a cubic domain of size one and advected by the following velocity field:

$$\begin{aligned} u &= 2 \sin(2\pi y) \sin^2(\pi x) \sin(2\pi z) \cos\left(\frac{\pi t}{T}\right) \\ v &= -\sin(2\pi x) \sin^2(\pi y) \sin(2\pi z) \cos\left(\frac{\pi t}{T}\right) \\ w &= -\sin(2\pi x) \sin^2(\pi z) \sin(2\pi y) \cos\left(\frac{\pi t}{T}\right) \end{aligned}$$

for a period $T = 3$. The CFL number is taken equal to 0.5. Due to the external velocity, the sphere undergoes significant thinning at the maximum deformation, making this test case challenging for low grid resolution. Figs. 8 and 9 show the deformed shape at $t = T/2$ and the final shape for, respectively, the grids $128 \times 128 \times 128$ and $256 \times 256 \times 256$. Consistent with the results presented above, the isosurface at $f = 0.5$ is shown. At the coarser resolution the volume fraction in the thinner part is lower than 0.5 which results in the empty area in the figure. The isosurface at a lower volume fraction, e.g., at 0.2, shows a continuous shape. In Table 4 the error E_1 is shown and compared with the one obtained by [45] using the Youngs normal calculation method [27] with a direction-splitting time integration scheme. The relative differences with [45] are within 20% and the convergence rate is comparable. The error in mass conservation on the coarsest grid $32 \times 32 \times 32$ and on the finest grid $256 \times 256 \times 256$ is, respectively, equal to 8.74×10^{-5} and 8.13×10^{-6} .

Summarising, we have shown that our implementation of a geometrical VOF algorithm is reliable by testing the solver gVoFoam on several 2D and 3D standard advection test cases. Both the comparisons with literature and with the software package ANSYS Fluent revealed a good agreement and generally a second order con-

Fig. 5. Contour level $f = 0.5$ for the 2D deformation field test on a 128×128 grid at the maximum deformation time $t = T/2$: (a) gVoFoam; (b) interFoam.

Fig. 6. Error E_1 for the 2D deformation field test as a function of the grid size h : the solid black line corresponds to gVoFoam; the solid blue line represents results of ANSYS Fluent; the dashed black line is the second order slope. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

vergence rate. Conversely, the experiments conducted on interFoam showed large errors and poor convergence of this method. In fact, the compression method of interFoam is based on a so

Fig. 8. 3D deformation field obtained with gVoFoam at a resolution of 128×128 : left $t = T/2$; right $t = T$.

called normalised variable diagram (NVD) scheme. Poor and oscillatory convergence behaviour of such NVD schemes was also found in [51] and is also expressed in the simulation of a rising bubble in this paper. In the next section we will turn our attention to a real-life problem of a rising bubble in a viscous liquid.

Fig. 7. 2D deformation field test: (a) computational time CT as a function of the number of grid points; (b) error E_1 as a function of the computational time CT . The solid lines show the results obtained with gVoFoam; the dashed lines correspond to the results obtained with interFoam.

Fig. 9. 3D deformation field obtained with gVoFoam at a resolution of $256 \times 256 \times 256$: left $t = T/2$; right $t = T$.

5. Rising bubble

In order to assess the accuracy and convergence of interFoam and gVoFoam on a fully coupled two-phase flow problem, in this section a simulation of a single rising bubble in a viscous liquid is considered. The full set of Eqs. (1), (2) and (5) is solved, first in 2D and subsequently also illustrated in 3D. A two-dimensional circular bubble of initial radius r_0 is simulated for 2 values of the Eötvös and Reynolds numbers, defined as

$$Eo = \frac{g \Delta \rho d_e^2}{\sigma}, \quad Re = \frac{\rho_l U_\infty d_e}{\mu_l}$$

where U_∞ is the terminal velocity of the bubble, ρ_l and μ_l the mass density and the dynamic viscosity of the liquid phase. The effective diameter d_e is defined as the diameter of a spherical bubble with the same volume as the actual bubble. The set up of the simulation is given in benchmark paper [15]. We reproduce the two test cases proposed in this benchmark reference, characterised as follows:

In the first test case the point (Re, Eo) in the bubble diagram of Grace [52] predicts a final ellipsoidal shape. In the second test case the buoyancy term is dominant with respect to the surface tension leading to a bubble shape with a strongly deformed ‘skirted’ interface. From the results presented in [15] we observe that the most sensible quantity to compare is the circularity, defined as [53]

$$c = \frac{P_a}{P_b} \quad (37)$$

where P_a is the perimeter of a circle having the same area as the bubble and P_b is the actual perimeter of the bubble. In order to quantify the quality of the numerical solution we adopt the same error measure as used in [15]. The L_1 norm of the error in the circularity is defined as

$$L_1 = \frac{\sum_n |c_{n,ref} - c_n|}{\sum_n |c_{n,ref}|} \quad (38)$$

where the summation $\sum_n$ is over all the time steps and the subscript ref indicates a reference grid. The benchmark presents results obtained from three different numerical codes: Transport Phenomena in 2D (TP2D) [54]; Free-Surface Library of Finite Element (FreeLIFE) [55]; Mathematics and object-oriented Numerics in MagDeburg (MoonMD) [56]. For test case 1, on the finest grids, no significant differences can be seen. We assume as a reference solution the finest grid and smallest time-step presented in the benchmark which is the one of the group TP2D. (Table 5)

We conduct a grid refinement study from a grid size $h = 1/40$ to $h = 1/320$. A time step Δt of $h/2$ is used in all the simulations, which corresponds to a maximum CFL number of approximately 0.35 for test case 1 and 0.26 for test case 2.

Table 5

Dimensionless parameters - rising bubble test cases.

Test case	Re	Eo	ρ_1/ρ_2	μ_1/μ_2
1	35	10	10	10
2	35	125	1000	100

5.1. Test case 1 ($Eo = 10, Re = 35$)

A key factor for the convergence of the numerical method, together with the advection algorithm described in Section 3, is an accurate evaluation of the surface tension that, in the model we use, depends on the smoothing length s_l . A too large value of s_l exaggerates softening of rapid changes in the bubble shape with consequent loss of accuracy. On the contrary, a too small value of s_l causes large errors in the computation of the second derivative of the volume fraction field, related to the curvature by (12). This leads to high spurious currents that deteriorate the solution. Fig. 10 shows a grid refinement study of the L_1 norm of the circularity for different smoothing lengths. In the left picture s_l is a function of the grid size, while in the right picture it is kept fixed despite the grid refinement.

In Fig. 10(a) we see that for very coarse grids the error is smaller when we use a smaller smoothing length. If we take, for example, the case where the smoothing length is quite large with $s_l = 8h$ at the coarser grid $h = 1/40$ we would have a value of the smoothing length equal to 0.2 which is comparable with the radius of the bubble equal to 0.25. In this situation the excessive smoothing will drive the bubble far from the reference solution. On the other hand, at higher resolutions a higher s_l/h gives more accurate results. If we keep increasing the grid resolution for small values of s_l/h , the convergence of the finite difference discretisation of (12) is not ensured since the higher derivatives in the truncation error can grow unbounded. This can be seen, for example, in the results for interFoam or gVoFoam with $s_l = 2h$. Fig. 10(b) shows that when the grid sufficiently resolves the length s_l second-order converge can be recovered.

The results obtained with $s_l = 4h$ are used for later comparisons. This value of the smoothing length allows to achieve a better accuracy in a wider range of grid resolutions and to preserve sharper corners in the bubble shape with respect to $s_h = 8h$. In Fig. 11 a comparison of the final shape obtained with gVoFoam at $s_l = 4h$ (a) and with interFoam (b) for the finest grid $h = 1/320$ is shown. The numerical shape obtained with interFoam on the finest grid substantially differs from the one of the benchmark while the numerical shape obtained with gVoFoam converges to the benchmark solution.

In Fig. 12(a) the computational time is shown as a function of the number of grid points. Differently from the advection test cases, the differences are small since most time is taken by the solution of the Poisson Eq. (9) in the Navier–Stokes solver, which is carried out in the same way for the two codes. This implies that a significant gain in accuracy can be obtained at relatively speaking no additional computational costs for the fully coupled problem. This is clearly visible in Fig. 12(b) where we plot the L_1 norm of the volume fraction field at the final time of the simulation as a function of the computational time. The points on the two curves have about the same values of CT while the ones corresponding to the results obtained with gVoFoam are shifted down towards higher accuracy.

Fig. 10. L_1 norm of the error in the circularity for different grid size h : (a) variable smoothing length; (b) fixed smoothing length. As reference we adopt the results as presented in [15].

Fig. 11. Bubble shape at time $t = 3$ for $h = 1/320$: (a) solution of gVoFoam (green line with diamonds); (b) solution of interFoam (green line with diamonds). In (a) and (b) we included the TP2D solution as point of reference (black line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 12. 2D rising bubble: (a) computational time (CT) as a function of the number of grid points; (b) L_1 norm of the volume fraction field as a function of the grid points. The dashed lines show the results obtained with gVoFoam using $s_l = 4h$; the solid lines correspond to results obtained with interFoam.

Fig. 13. Computed bubble shapes for test case 2 on a grid 640×1280 : (a) $t = 0$; (b) $t = 1.5$; (c) $t = 3$.

Fig. 14. (a) Detail of the final bubble shape for different grid refinement obtained with gVoFoam at $s_l = 4h$; (b) comparison of the final shape with the benchmark grid reference results.

5.2. Test case 2 ($Eo = 125, Re = 35$)

This second test case shows a considerably higher deformation of the interface. For high Eötvös numbers the buoyancy term is dominant compared to the surface tension, causing the appearance of significant wakes. Fig. 13 shows the computed bubble shapes at different times. The progressive thinning of the skirts requires a high resolution in order to be captured well. For the reasons explained in the previous section we choose a smoothing length $s_l = 4h$, creating a good balance between accuracy and computing time. Fig. 14(a) shows a detail of the final shape for different grid resolutions and Fig. 14(b) a comparison of the final shape obtained with the finest grid from the three codes in the benchmark [15].

In the region far away from the skirt the computed shapes overlap very closely. The occurrence of break up, as observed in the solution of TP2D, is dependent on the grid resolution and on the numerical method used for tracking of bubble. As can be inferred, even at the higher resolutions the details that appear in the shape are captured by just a few grid cells, hence leaving still quite some contribution of spatial discretization error on such finer details. The wake obtained from gVoFoam is comparable in length and thickness with the ones obtained from the benchmark. At

Fig. 15. 3D rising bubble on a grid $160 \times 320 \times 160$ with the corresponding velocity field (arrows) on a plane passing through the centre of mass of the bubble: (a) $t = 1$; (b) $t = 3$. One velocity vector in each 10 grid points is shown. The magnitude of the velocity is shown on the background. The centre of the bubble is initially placed at $(0.5, 0.5, 0.5)$

$h = 1/640$ we have approximately 4 cells across the thinnest point of the tail and no break up occurs with our method. The solver gVoFoam is thus shown to be able to capture highly deformed bubble shapes.

5.3. 3D rising bubble

To show the capability of gVoFoam in 3D we perform a simulation of an initially spherical rising bubble placed at $(0.5, 0.5, 0.5)$ under the same physical parameters as in test case 2. Fig. 15 shows the results at $t = 1$ and $t = 3$, where the iso-surface of the volume fraction at 0.5 is shown together with the velocity field on a slice passing through the centre of mass of the bubble. Fig. 16 shows the same results for a spherical bubble initially displaced off-centre at $(0.625, 0.5, 0.625)$. For the sake of clarity, only one velocity vector per 10 grid points is shown. The colour map indicates the magnitude of the velocity field.

In Fig. 17 the shape of the bubble, corresponding to the symmetric case, in the vertical mid plane is shown for different grid refinements at $t = 1$ and $t = 3$. The point $Eo = 125$ and $Re = 35$ in

Fig. 16. 3D rising bubble on a grid $160 \times 320 \times 160$ with the corresponding velocity field (arrows) on a plane passing through the centre of mass of the bubble: (a) $t = 1$; (b) $t = 3$. One velocity vector in each 10 grid points is shown. The magnitude of the velocity is shown on the background. The centre of the bubble is initially placed at $(0.625, 0.5, 0.625)$

the Grace diagram falls in the region between dimpled and skirted bubble shape. This behaviour is well reproduced by gVoFoam as evidenced from the mid-plane shape at $t = 3$. A more quantitative convergence can be deduced from Fig. 17(b) by computing the volume fraction field on successive grid refinements. Computing $E_f = \sum_i |f_i(h) - f_i(h/2)| / |f_i(h/2) - f_i(h/4)|$ with $h = 1/40$ and i the index of the grid points, we find a value equal to 3.9 which suggests second order convergence. The results on the finer grids are linearly interpolated on the grid $h = 1/40$ so that the index i has the same bounds.

The simulation on the finest grid $160 \times 320 \times 160$ has been carried by using 32 cores of 2.6 GHz Intel Xeon Processors, with a CPU time of 2.6 h.

6. Conclusion

In this paper we considered multiphase flow and compared two numerical methods for the simulation of air bubbles in a liquid. In particular, the compression method [6] and the geometric method [2,5]. The combination of a VOF-PLIC algorithm and a smoothing

technique for the computation of the surface tension has been shown to give accurate results and convergence under grid refinement. In order to assess the methods we performed a number of advection tests in which interfaces are transported in an assumed external velocity field. A quantitative comparison with literature has been made. In addition, the motion of bubbles in a liquid flow was simulated by solving the Navier-Stokes equations coupled to the volume fraction field equation.

From the advection test cases, where the volume fraction equation is solved, we observed that the compression method, implemented in the solver interFoam, fails qualitatively. In contrast, the results obtained with the new gVoFoam solver, based on the VOF-PLIC algorithm, adhere closely to literature and second order convergence was found in the 2D deformation field test case [24]. The VOF-PLIC algorithm is not exactly mass preserving but the effect is relatively small for the coarsest grid and decreases about one order of magnitude when halving the mesh size. On the other hand, the compression method is mass conserving up to machine precision. On these advection test cases interFoam was found to be three times faster than gVoFoam for the finest grid resolution, but the latter is about two order of magnitude more accurate than the former. An independent test performed with the software package ANSYS Fluent provided an additional confirmation of the reliability of the gVoFoam implementation.

For the fully coupled problem of a rising bubble in a viscous liquid, the solver gVoFoam gives results in good agreement with a literature benchmark [15]. Furthermore, second order convergence was achieved by using a smoothing length numerically well resolved by the computational grid. This allows an accurate computation of the surface tension and a reduction of spurious currents. On the contrary, the solver interFoam does not show convergence under grid refinement at all and the results on the finest grid substantially differ from the one of the benchmark. Differently from the advection test cases, only a small relative difference in the computational time has been observed, due to the fact that most time is taken by the solution of the pressure equation. This implies that more accurate results can be obtained at virtually no extra computational costs for the fully coupled problem. Finally, the capability of gVoFoam has been shown on a 3D simulation of a rising bubble where an error measure on the volume fraction field suggests second order converge in the range of grids analysed.

The solver gVoFoam has been implemented in the open-source software OpenFOAM®. At the current state it is limited to Cartesian

Fig. 17. Bubble shape on a vertical mid plane on the grids $40 \times 80 \times 40$ (green line), $80 \times 160 \times 80$ (red line), $160 \times 320 \times 160$ (black line) at: (a) $t = 1$; (b) $t = 3$. The results are obtained with gVoFoam using a smoothing length equal to $4h$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

grids. The next challenge will be the extension of gVoFoam with phase change, to which future research will be devoted.

Acknowledgements

This research is supported by the Dutch Technology Foundation STW, applied-science division of NWO (Netherlands Organization for Scientific Research), and the Technology Program of the Ministry of Economic Affairs of the Netherlands. This work was carried out on the Dutch national e-infrastructure with the support of SURF Cooperative.

Appendix A

Relations (22) and (23) have been implemented in OpenFOAM® as well as the time splitting advection algorithm and the geometrical evaluation of the fluxes explained in Section 3.3. The functions $\mathcal{T}$ and $\mathcal{T}^{-1}$ are derived, for a unit cube, by Scardovelli and Zaleski [32], and here reported with additional details about the implementation in OpenFOAM® for a parallelepiped. Consider the generic cell of Fig. A.18 with edges Δ_1 , Δ_2 and Δ_3 , that is cut by the plane of Eq. (21)

$$n_1 x_1 + n_2 x_2 + n_3 x_3 = q \quad (\text{A.1})$$

where n_1 , n_2 and n_3 are the components of the normal vector to the plane. The intersection volume below the plane be calculated by boolean subtraction operations on the volume of the hexahedron ABCDEFGH. In particular, the volume ABGHKNML is found to be

$$\begin{aligned} \text{Volume(ABGHKNML)} &= \text{Volume(AJIK)} - \text{Volume(HPIL)} \\ &\quad - \text{Volume(BJON)} + \text{Volume(GPOM)} \end{aligned} \quad (\text{A.2})$$

whose analytical expression is given by

$$\begin{aligned} \text{Volume(ABGHKNML)} &= \frac{1}{6n_1 n_2 n_3} [q^3 - (q - n_1 \Delta_1)^3 - (q - n_2 \Delta_2)^3 \\ &\quad + (q - n_1 \Delta_1 - n_2 \Delta_2)^3] \end{aligned} \quad (\text{A.3})$$

This relation can be generalised to

Fig. A.18. Geometry of a cell cut by a plane.

$$\begin{aligned} V &= \frac{1}{6n_1 n_2 n_3} \left[q^3 - \sum_{i=1}^3 H(q - n_i \Delta_i) (q - n_i \Delta_i)^3 \right. \\ &\quad \left. + \sum_{i=1}^3 H(q - q_{\max} + n_i c_i) (q - q_{\max} + n_i c_i)^3 \right] \end{aligned} \quad (\text{A.4})$$

where $H(\cdot)$ is the Heaviside function and $q_{\max} = \sum_{i=1}^3 n_i \Delta_i$. Expression (A.4) is conceptually easy, and can be applied to every possible case of a plane cutting a hexahedron. However, this results in a large number of combinations, each of them having a different form of (A.4). The complexity of the problem can be reduced by making use of parametrisation and symmetries. We first use the following change of variables:

$$\begin{aligned} x'_1 &= x_1 / \Delta_1 \\ x'_2 &= x_2 / \Delta_2 \\ x'_3 &= x_3 / \Delta_3 \end{aligned} \quad (\text{A.5})$$

so that in the transformed space (x'_1, x'_2, x'_3) the cell is a unit cube cut by the plane of equation

$$n'_1 x'_1 + n'_2 x'_2 + n'_3 x'_3 = q \quad (\text{A.6})$$

and the volume V in (A.4) becomes the volume fraction. It is convenient to have all three components of the normal vector positive. To do so, we can apply a mirror reflection around the plane $\Delta_m/2$ since the resulting volume fraction will be the same. Here, m is the index of the component of the normal vector that has a negative value. In formulas we have:

$$\begin{aligned} n'_i &= |n_i| \\ q' &= q + \sum_{i=1}^3 \max(0, -n'_i) \end{aligned} \quad (\text{A.7})$$

A scalar quantity is also invariant under any permutation of the coordinate system, so we can sort the normal components:

$$n'_3 < n'_2 < n'_1 \quad (\text{A.8})$$

The last step is to scale Eq. (A.6) by the factor $\sum_{i=1}^3 n'_i$:

$$\frac{n'_1 x'_1}{n'_1 + n'_2 + n'_3} + \frac{n'_2 x'_2}{n'_1 + n'_2 + n'_3} + \frac{n'_3 x'_3}{n'_1 + n'_2 + n'_3} = \frac{q'}{n'_1 + n'_2 + n'_3}$$

i.e.,

$$n''_1 x'_1 + n''_2 x'_2 + n''_3 x'_3 = q'' \quad (\text{A.9})$$

which leads to $q''_{\max} = 1$. With these transformations both V and q vary in the range $[0, 1]$ and the function $V = f(q)$ has an odd symmetry at $q = 1/2$, so that we can restrict the analysis to the range $[0 \leq q \leq 1/2]$. It can be shown that the complexity of the problem is reduced to 5 cases. If we define $d = 6n_1 n_2 n_3$, $n_{12} = n_1 + n_2$ and $n = \min(n_1 + n_2, n_3)$, then the expressions for the volume fraction are:

$$\begin{aligned} V &= \frac{q^3}{d} & 0 \leq q < n_1 \\ V &= \frac{q(q - n_1)}{2n_2 n_3} + \frac{n_1^3}{d} & n_1 \leq q < n_2 \\ V &= \frac{q^2(3n_{12} - q) + n_1^2(n_1 - 3q) + n_2^2(n_2 - 3q)}{d} & n_2 \leq q < n \end{aligned}$$

and for the last interval depending whether $n = n_3 < n_{12}$ or $n = n_{12} < n_3$ we have the following two possibilities:

$$\begin{aligned} V &= \frac{q^2(3 - 2q) + n_1^2(n_1 - 3q) + n_2^2(n_2 - 3q) + n_3^2(n_3 - 3q)}{d} & n_3 \leq q < 1/2 \\ V &= \frac{2q - n_{12}}{2n_3} & n_{12} \leq q < 1/2 \end{aligned}$$

The function $\mathcal{T}^{-1}$ can be derived in an analogous way. The transformations (A.5), (A.8) and (A.9) are applied to the initial problem (A.4). The function $q = f(V)$ is then given by

$$\begin{aligned} q &= \sqrt[3]{Vd} & 0 \leq V < V_1 \\ q &= \frac{1}{2} \left(n_1 + \sqrt{n_1^2 + 8n_2n_3(V - V_1)} \right) & V_1 \leq V < V_2 \\ a_3q^3 + a_2q^2 + a_1q + a_0 &= 0 & V_2 \leq q < V_3 \end{aligned}$$

and again for the last interval, depending on whether $V_3 = V_{31} < V_{32}$ or $V_3 = V_{32} < V_{31}$, we have the following two possibilities:

$$\begin{aligned} a_3'q^3 + a_2'q^2 + a_1'q + a_0' &= 0 & V_{31} \leq V \leq 1/2 \\ q &= n_3V + \frac{n_{12}}{2} & V_{32} \leq V \leq 1/2 \end{aligned}$$

where $V_1 = n_1^3/d$, $V_2 = (n_2 - n_1)/(2n_3) + V_1$ and $V_3 = [n_3^2(3n_{12} - n_3) + n_1^2(n_1 - 3n_3) + n_2^2(n_2 - 3n_3)]/d$ if $n_3 < n_{12}$ else $V_3 = n_{12}/(2n_3)$. The coefficients of the first cubic polynomial are: $a_3 = 1$, $a_2 = -3n_{12}$, $a_1 = 3(n_1^2 + n_2^2)$, $a_0 = -(n_1^2 + n_2^2 + Vd)$; the coefficients of the second cubic polynomial are: $a_3' = 1$, $a_2' = -3/2$, $a_1' = (3/2)(n_1^2 + n_2^2 + n_3^3)$, $a_0' = -(n_1^3 + n_2^3 + n_3^3)/2 + d/2$. The roots of the polynomials corresponding to the proper value of the intercept q have been worked out by Scardovelli [32]. The solution of the first polynomial is given by

$$q = \sqrt{p_0} \left(\sqrt{3 \sin(\theta)} - \cos(\theta) + n_{12} \right)$$

where $p_0 = 2n_1n_2$, $\theta = \arccos \left(q_0 / \left(p_0^{3/2} \right) \right) / 3$ with $q_0 = 3n_1n_2(n_{12}/2 - n_3V)$. The solution of the second polynomial is equal to

$$q = \sqrt{p_0} \left(\sqrt{3 \sin(\theta)} - \cos(\theta) + 1/2 \right)$$

where $p_0 = n_1(n_2 + n_3) + n_2n_3 - 1/4$, $\theta = \arccos \left(q_0 / \left(p_0^{3/2} \right) \right) / 3$ with $q_0 = (3/4)n_1n_2n_3(1 - 2V)$. Finally, the inverse of transformation (A.7) is used to transform the plane back to its actual position.

The function $\mathcal{T}^{-1}$ reconstructs the interface at the beginning of each advection step in (24) and (25). By means of the function $\mathcal{T}$ we compute the geometrical fluxes. Depending on the sign of the velocity, the flux will leave or enter the considered cell. Since we use a splitting advection algorithm we can analyse only one direction at the time. For example, we can consider the generic cell i, j, k and the split direction x_1 . When at the face $(i - 1/2, j, k)$ we have $u < 0$ a portion of volume of phase 1 will leave the cell. The equation of the plane that describes the interface is defined in a coordinate system centred in a vertex of the cell (i, j, k) so we can apply $\mathcal{T}$ to $(q_{i,j,k}, n_{i,j,k})$. On the other hand, when $u > 0$ a portion of volume of phase 1 will enter the cell (i, j, k) from the cell $(i - 1, j, k)$. The equation of the piece of planar interface that will be convected is now defined in a coordinate system centred in a vertex of the cell $(i - 1, j, k)$. The change of coordinates

$$x_1 = (\Delta_1 - u\Delta t) + \bar{x}_1$$

will bring back the problem to the standard configuration depicted in Fig. A.18.

The 2D and 3D algorithms have been implemented in a new solver class VOF, where the advection directions are cyclicly alternated as explained in Section 3.3. Given the directional nature of the splitting time integration method a new mesh topology class, called vofMesh, was created. This class contains a list of cells that have as attribute a list of ordered faces along the 3 orthogonal directions. This feature, not present in OpenFOAM®, enables a more efficient implementation of the geometrical VOF method. The parallel implementation of the algorithm is performed with the standard interface MPI [57].

References

- [1] Hirt CW, Nichols BD. Volume of fluid (VOF) method for the dynamics of free boundaries. *J Comp Phys* 1981;39(1):201–25. doi:10.1016/0021-9991(81)90145-5.
- [2] Rider WJ, Kothe DB. Reconstructing volume tracking. *J Comp Phys* 1998;141(2):112–52. doi:10.1006/jcph.1998.5906.
- [3] Scardovelli R, Zaleski S. Direct numerical simulation of free-surface and interfacial flow. *Ann Rev Fluid Mech* 1999;31(1):567–603. doi:10.1146/annurev.fluid.31.1.567.
- [4] Popinet S. An accurate adaptive solver for surface-tension-driven interfacial flows. *J Comp Phys* 2009;228(16):5838–66. doi:10.1016/j.jcp.2009.04.042.
- [5] Puckett EG, Almgren AS, Bell JB, Marcus DL, Rider WJ. A high-order projection method for tracking fluid interfaces in variable density incompressible flows. *J Comp Phys* 1997;130(2):269–82. doi:10.1006/jcph.1996.5590.
- [6] Rusche H. Computational fluid dynamics of dispersed two-phase flows at high phase fractions. Imperial College London (University of London); 2003. Ph.D. thesis.
- [7] Unverdi SO, Tryggvason G. A front-tracking method for viscous, incompressible, multi-fluid flows. *J Comp Phys* 1992;100(1):25–37. doi:10.1016/0021-9991(92)90307-K.
- [8] Ye T, Shyy W, Chung JN. A fixed-grid, sharp-interface method for bubble dynamics and phase change. *J Comp Phys* 2001;174(2):781–815. doi:10.1006/jcph.2001.6938.
- [9] van Sint Annaland M, Deen N, Kuipers J. Numerical simulation of gas bubbles behaviour using a three-dimensional volume of fluid method. *Chem Eng Sci* 2005;60(11):2999–3011. doi:10.1016/j.ces.2005.01.031.
- [10] Esmarelli A, Tryggvason G. A direct numerical simulation study of the buoyant rise of bubbles at $o(100)$ reynolds number. *Phys Fluids* (1994-present) 2005;17(9):093303. doi:10.1063/1.2056617.
- [11] Tryggvason G, Esmarelli A, Lu J, Biswas S. Direct numerical simulations of gas/liquid multiphase flows. *Fluid Dyn Res* 2006;38(9):660–81. doi:10.1016/j.fluid.2005.08.006.
- [12] Bonometti T, Magnaudet J. An interface-capturing method for incompressible two-phase flows. validation and application to bubble dynamics. *Int J Multiph Flow* 2007;33(2):109–33. doi:10.1016/j.ijmultiphaseflow.2006.07.003.
- [13] Klostermann J, Schaake K, Schwarze R. Numerical simulation of a single rising bubble by VOF with surface compression. *Int J Num Meth Fluids* 2013;71(8):960–82. doi:10.1002/fld.3692.
- [14] OpenCFD Ltd. User and programmer's guide; 2013. URL <http://www.openfoam.com>.
- [15] Hysing S, Turek S, Kuzmin D, Parolini N, Burman E, Ganesan S, et al. Quantitative benchmark computations of two-dimensional bubble dynamics. *Int J Num Meth Fluids* 2009;60(11):1259–88. doi:10.1002/fld.1934.
- [16] Francois MM, Cummins SJ, Dendy ED, Kothe DB, Sicilian JM, Williams MW. A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework. *J Comp Phys* 2006;213(1):141–73. doi:10.1016/j.jcp.2005.08.004.
- [17] Renardy Y, Renardy M. Prost: a parabolic reconstruction of surface tension for the volume-of-fluid method. *J Comp Phys* 2002;183(2):400–21. doi:10.1006/jcph.2002.7190.
- [18] Brackbill J, Kothe DB, Zemach C. A continuum method for modeling surface tension. *J Comp Phys* 1992;100(2):335–54. doi:10.1016/0021-9991(92)90240-Y.
- [19] Baltussen M, Kuipers J, Deen N. A critical comparison of surface tension models for the volume of fluid method. *Chem Eng Sci* 2014;109:65–74.
- [20] Zalesak ST. Fully multidimensional flux-corrected transport algorithms for fluids. *J Comp Phys* 1979;31(3):335–62. doi:10.1016/0021-9991(79)90051-2.
- [21] Sweby PK. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM J Numer Anal* 1984;21(5):995–1011. doi:10.1137/0721062.
- [22] Leonard B. Simple high-accuracy resolution program for convective modelling of discontinuities. *Int J Numer Methods Fluids* 1988;8(10):1291–318.
- [23] Leonard B. The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection. *Comput Methods Appl Mech Eng* 1991;88(1):17–74. doi:10.1016/0045-7825(91)90232-U.
- [24] Rudman M. Volume-tracking methods for interfacial flow calculations. *Int J Num Meth Fluids* 1997;24(7):671–91.
- [25] Jasak H, Weller H, Gosman A. High resolution nvd differencing scheme for arbitrarily unstructured meshes. *Int J Num Meth Fluids* 1999;31(2):431–49.
- [26] Noh WF, Woodward P. SLIC (simple line interface calculation). In: Proceedings of the fifth international conference on numerical methods in fluid dynamics June 28–July 2, 1976 Twente University, Enschede. Springer; 1976. p. 330–40. doi:10.1007/3-540-08004-X_336.
- [27] Youngs DL. Time-dependent multi-material flow with large fluid distortion. *Numer Methods Fluid Dyn* 1982;24:273–85.
- [28] Leonard B, Lock A, MacVean M. Conservative explicit unrestricted-time-step multidimensional constancy-preserving advection schemes. *Monthly Weather Rev* 1996;124(11):2588–606.
- [29] Maric T, Marschall H., Bothe D. voFoam-a geometrical volume of fluid algorithm on arbitrary unstructured meshes with local dynamic adaptive mesh refinement using openfoam. *arXiv preprint arxiv.org/abs/1305.3417*.
- [30] Scardovelli R, Zaleski S. Interface reconstruction with least-square fit and split eulerian-lagrangian advection. *Int J Num Meth Fluids* 2003;41(3):251–74. doi:10.1002/fld.431.

- [31] Sussman M, Puckett EG. A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows. *J Comp Phys* 2000;162(2):301–37. doi:[10.1006/jcph.2000.6537](https://doi.org/10.1006/jcph.2000.6537).
- [32] Scardovelli R, Zaleski S. Analytical relations connecting linear interfaces and volume fractions in rectangular grids. *J Comp Phys* 2000;164(1):228–37. doi:[10.1006/jcph.2000.6567](https://doi.org/10.1006/jcph.2000.6567).
- [33] Inc. A. ANSYS Fluent 12.0 user's guide; URL <http://www.ansys.com>.
- [34] Prosperetti A, Tryggvason G. *Computational methods for multiphase flow*. Cambridge University Press; 2007.
- [35] Jasak H. *Error analysis and estimation for the finite volume method with applications to fluid flows*. Imperial College London (University of London); 1996. Ph.D. thesis.
- [36] Rhie C, Chow W. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA J* 1983;21(11):1525–32. doi:[10.2514/3.8284](https://doi.org/10.2514/3.8284).
- [37] Issa RI. Solution of the implicitly discretised fluid flow equations by operator-splitting. *J Comp Phys* 1986;62(1):40–65. doi:[10.1016/0021-9991\(86\)90099-9](https://doi.org/10.1016/0021-9991(86)90099-9).
- [38] Courant R, Friedrichs K, Lewy H. Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen* 1928;100(1):32–74. doi:[10.1007/BF01448839](https://doi.org/10.1007/BF01448839).
- [39] Lafaurie B, Nardone C, Scardovelli R, Zaleski S, Zanetti G. Modelling merging and fragmentation in multiphase flows with surfer. *J Comp Phys* 1994;113(1):134–47. doi:[10.1006/jcph.1994.1123](https://doi.org/10.1006/jcph.1994.1123).
- [40] Popinet S, Zaleski S. A front-tracking algorithm for accurate representation of surface tension. *Int J Num Meth Fluids* 1999;30(6):775–93.
- [41] Harvie DJ, Davidson M, Rudman M. An analysis of parasitic current generation in volume of fluid simulations. *Appl Math Model* 2006;30(10):1056–66. doi:[10.1016/j.apm.2005.08.015](https://doi.org/10.1016/j.apm.2005.08.015).
- [42] Hoang DA, van Steijn V, Portela LM, Kreutzer MT, Kleijn CR. Benchmark numerical simulations of segmented two-phase flows in microchannels using the Volume of Fluid method. *Comput Fluids* 2013;86:28–36.
- [43] Cummins SJ, Francois MM, Kothe DB. Estimating curvature from volume fractions. *Comput Structures* 2005;83(6):425–34. doi:[10.1016/j.compstruc.2004.08.017](https://doi.org/10.1016/j.compstruc.2004.08.017).
- [44] Berberović E, van Hinsberg NP, Jakirlić S, Roisman IV, Tropea C. Drop impact onto a liquid layer of finite thickness: Dynamics of the cavity evolution. *Phys Rev E* 2009;79(3):036306.
- [45] Liovic P, Rudman M, Liow J-L, Lakehal D, Kothe D. A 3d unsplit-advection volume tracking algorithm with planarity-preserving interface reconstruction. *Comput Fluids* 2006;35(10):1011–32. doi:[10.1016/j.compfluid.2005.09.003](https://doi.org/10.1016/j.compfluid.2005.09.003).
- [46] Pilliod JE, Puckett EG. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *J Comp Phys* 2004;199(2):465–502. doi:[10.1016/j.jcp.2003.12.023](https://doi.org/10.1016/j.jcp.2003.12.023).
- [47] LeVeque RJ. High-resolution conservative algorithms for advection in incompressible flow. *SIAM J Numer Anal* 1996;33(2):627–65. doi:[10.1137/0733033](https://doi.org/10.1137/0733033).
- [48] Enright D, Fedkiw R, Ferziger J, Mitchell I. A hybrid particle level set method for improved interface capturing. *J Comp Phys* 2002;183(1):83–116. doi:[10.1006/jcph.2002.7166](https://doi.org/10.1006/jcph.2002.7166).
- [49] Hernández J, López J, Gómez P, Zanzi C, Faura F. A new volume of fluid method in three dimensions—part i: Multidimensional advection method with face-matched flux polyhedra. *Int J Num Meth Fluids* 2008;58(8):897–921. doi:[10.1002/flid.1776](https://doi.org/10.1002/flid.1776).
- [50] Jemison M, Loch E, Sussman M, Shashkov M, Arienti M, Ohta M, et al. A coupled level set-moment of fluid method for incompressible two-phase flows. *J Sci Comput* 2013;54(2–3):454–91. doi:[10.1007/s10915-012-9614-7](https://doi.org/10.1007/s10915-012-9614-7).
- [51] Zhu J. On the higher-order bounded discretization schemes for finite volume computations of incompressible flows. *Comput Methods in Appl Mech Eng* 1992;98(3):345–60. doi:[10.1016/0045-7825\(92\)90003-3](https://doi.org/10.1016/0045-7825(92)90003-3).
- [52] Grace J. Shapes and velocities of bubbles rising in infinite liquids. *Trans Inst Chem Eng* 1973;51(2):116–20.
- [53] Wadell H. Sphericity and roundness of rock particles. *J Geol* 1933;310–31.
- [54] Turek S. *Efficient solvers for incompressible flow problems: An algorithmic and computational approach*, 6. Springer Science & Business Media; 1999.
- [55] Parolini N, Burman E. A finite element level set method for viscous free-surface flows. *Ser Adv Math Appl Sci* 2005;69:416.
- [56] John V, Matthies G. MoonMD—a program package based on mapped finite element methods. *Comput Visual Sci* 2004;6(2–3):163–70. doi:[10.1007/s00791-003-0120-1](https://doi.org/10.1007/s00791-003-0120-1).
- [57] Gropp W, Lusk E, Doss N, Skjellum A. A high-performance, portable implementation of the MPI message passing interface standard. *Parallel Comput* 1996;22(6):789–828. doi:[10.1016/0167-8191\(96\)00024-5](https://doi.org/10.1016/0167-8191(96)00024-5).