Computing interface curvature from volume fractions: A hybrid approach

H.V. Patel\textsuperscript{a}, J.A.M. Kuipers\textsuperscript{a}, E.A.J.F. Peters\textsuperscript{a,}\textsuperscript{*}

\textsuperscript{a}Multiphase Reactors Group, Department of Chemical Engineering and Chemistry, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands.

Abstract

The Volume of Fluid method is extensively used for the multiphase flows simulations in which the interface between two fluids is represented by a discrete and abruptly-varying volume fractions field. The Heaviside nature of the volume fractions field presents an immense challenge for the accurate computation of the interface curvature and induces the spurious velocities in the flows with surface-tension effects. A 3D hybrid approach is presented combining the Convolution and Generalized Height Function method for the curvature computation. The volumetric surface tension forces are computed using the balanced-force continuum surface force model. It provides a high degree of robustness at lower grid resolutions with first-order convergence and high accuracy at higher grid resolutions with second-order convergence. The present method is validated for several test cases including a stationary droplet, an oscillating droplet and the buoyant rise of gas bubbles over a wide range of Eötvös (\textit{Eo}) and Morton (\textit{Mo}) numbers. Our computational results show an excellent agreement with analytical/experimental results with desired convergence behavior.

\textit{Keywords:} Volume of Fluid, Hybrid interface curvature, Balanced-force, Convolution, Generalized height function

1. Introduction

Multiphase flows involving fluid-fluid interfaces are ubiquitous in nature and technology. Rain drops, atomizing water jet, spray driers, emulsions, bubble swarms and trickle bed reactors are just a few examples. Typically these flows have high density and/or viscosity ratios with high surface tension effects which tend to produce topologically complex and dynamically evolving interfaces. So, the accurate numerical simulation of such flows has attracted a lot of attention among researchers.

Numerical simulations of multiphase flows need to address two primary challenges: i) mass-conserving advection of the fluid phases and ii) accurate computation of the surface tension forces at the fluid-fluid interface. A wide range of multi-fluid interface tracking/capturing methods have been developed to simulate multiphase flows and they mainly differ with respect to the way they tackle aforementioned challenges. A holistic overview of these methods is presented by Scardovelli and Zaleski (1999). Examples include the Volume of Fluid (Hirt and Nichols, 1981), Level Set (Sussman et al., 1994), Front-Tracking (Tryggvason et al., 2001), multiphase Lattice-Boltzmann (Li et al., 2016) and moving-grids (Takagi et al., 1997) methods.

The volume of fluid (VOF) method uses a discrete and abruptly-varying volume fractions (\textit{F}) field to represent the fractional amount of a reference fluid present in each computational cell. Advection of \textit{F} is treated by a pseudo-Lagrangian geometrical advection schemes to minimize numerical diffusion. Interface
reconstruction is required for geometrical advection of $F$ which is done by piecewise linear interface calculation (PLIC) following Youngs (1982). Weymouth and Yue (2010) shows that the VOF method can be 'exactly' mass conservative however negligible mass errors may arise due to finite machine precision during numerical computations.

The volumetric surface tension force is computed using the continuum surface force (CSF) model proposed by Brackbill et al. (1992). CSF model suggests that the accuracy of the surface tension force is mostly determined by the computed local interface curvature. Accurate computation of the curvature is particularly difficult due to the discontinuous nature of $F$ field. Inaccurate curvature, however, along with the improper discretization of the surface tension force creates an imbalance of pressure and surface tension forces at the interface which induces spurious velocities in the flow field. A detailed analysis of generation and scaling of spurious velocities in VOF simulations can be found in Harvie et al. (2006). So, to reduce/eliminate these spurious velocities two problems need to be tackled distinctly: i) surface tension and pressure forces need to be discretized at the same location (balanced-force concept by Francois et al. (2006)) and ii) accurate computation of the interface curvature.

Interface curvature can be computed as a second order spatial derivative of the abruptly-varying $F$ field. Different methods are available for the numerical computation of interface curvature and an excellent comparison is presented by Cummins et al. (2005); Francois et al. (2006). The Convolution (CV) method uses a smoothing kernel to convolute the $F$ field before differentiating it. This removes the high frequency aliasing error which otherwise would occur due to the numerical differentiation of $F$ (Heaviside function). Accuracy and convergence of the CV method depends upon the length of the smoothing kernel. The Height Function (HF) method computes the heights by summing the $F$ field across the interface which produces differentiable heights from an abruptly-varying $F$ field. A Standard Height Function (SHF) method uses a fixed $7 \times 3$ stencil in 2D ($7 \times 3 \times 3$ in 3D) for this summation process. A Generalized Height Function (GHF) method (Popinet, 2009) uses an adaptive stencil to compute heights which produces a better curvature estimate especially in the case of complex topologies and interface merging/breakup. Reconstructed distance function (RDF) method computes a distance function from the interface in the VOF framework. This distance function is smoothly varying and hence curvature is directly computed from the numerical differentiation of the same. RDF method is similar to the one used for the coupled VOF-Level Set simulations by Sussman and Puckett (2000). A Least Squared (LS) method is another option which computes the curvature by fitting a parabola in 2D (paraboloid in 3D) either on the volume fractions directly (Renardy and Renardy, 2002) or on a set of interfacial points (Popinet, 2009).

Only a few attempts have been made to develop hybrid methods for curvature computation. Popinet (2009) presents a hybrid method on adaptive grids which uses the GHF method at higher grid resolutions and the LS method at lower grid resolutions. The LS method is computationally expensive and requires a complex implementation compared to other methods. Owkes and Desjardins (2015) presents a mesh decoupled HF method where HF stencils are constructed in the direction of the interface normal which may not be aligned with the underlying Eulerian grid. Embedded height-function technique by Ivey and Moin (2015) constructs HF stencils on unstructured non-convex polyhedral meshes. This method embeds traditional HF stencils in the unstructured mesh and geometrically interpolates the volume fraction information from the mesh to the HF stencil. Both of these methods uses fixed length HF stencils which drastically reduces the curvature accuracy during interface merging or breakup.

Cummins et al. (2005) concluded that the CV or RDF method exhibits a greater robustness over the HF method and doesn’t breakdown catastrophically when the interfacial length scale (radius of curvature) is
not adequately resolved by the grid. On the other hand, the HF method computes very accurate curvatures which converges with second-order when the interface is adequately resolved. So, the choice of appropriate curvature finding method depends on the number of grid cells across the radius of curvature. They called for a hybrid/unified curvature finding method which uses a robust CV or RDF method at lower and an accurate HF method at higher grid resolutions. Such a hybrid method should have a branching algorithm to decide which curvature finding method is to be used based on the local interface topology.

In this paper, a 3D hybrid approach (CV-GHF) is presented which combines the CV and GHF methods for the curvature computation. The volumetric surface tension force is computed using the balanced-force CSF model. The present method computes highly accurate curvatures with second-order convergence at higher grid resolutions. A branching algorithm is an in-built part of the present method which automatically switches the curvature computation method from GHF to CV at lower grid resolutions. The method is quite robust and shows first-order convergence at lower grid resolutions. The paper is organized as follows: First, we describe the governing Navier-Stokes equations for multiphase flows along with the balanced-force CSF model in VOF framework. Subsequently, we discuss the CV, HF and hybrid CV-GHF method with the branching algorithm in detail. Next, we lay down different scenarios where the GHF method exhibits better performance than the SHF method. Curvature errors are then quantified for the present CV-GHF method to check the convergence behavior. Finally, an extensive validation of the method is reported with different test cases including a stationary droplet, an oscillating droplet and the buoyant rise of gas bubbles over a wide range of physical properties of the gas-liquid system.

2. Numerical method

In this section the governing equations for the multiphase flow are described along with the discretization and solution methodology in brief. The main focus will be based on the balanced-force concept which is used to discretize the surface tension force in present work.

2.1. Governing equations

For the incompressible, unsteady, Newtonian multiphase flows the mass and momentum conservation (Navier-Stokes) equations can be represented in terms of the single velocity field \( \mathbf{u} \) formalism:

\[
\nabla \cdot \mathbf{u} = 0
\]

\[
\rho \frac{\partial \mathbf{u}}{\partial t} + \rho \nabla \cdot (\mathbf{uu}) = -\nabla p + \nabla \cdot \tau + \rho \mathbf{g} + \mathbf{F}_\sigma
\]

where \( \tau = \mu [\nabla \mathbf{u} + (\nabla \mathbf{u})^T] \) is the fluid stress tensor, \( \mathbf{g} \) is the acceleration due to gravity and \( \mathbf{F}_\sigma \) is the local surface tension force accounting for the presence of curved deformable fluid-fluid interfaces. The local averaged density \( (\rho) \) is evaluated by the linear averaging of the densities of the individual fluid phases:

\[
\rho = F \rho_1 + (1 - F) \rho_2
\]

Note that we use volume fraction \( F = 1 \) and \( 0 \) for the computational cells fully occupied by fluid 1 and fluid 2, respectively. \( 0 < F < 1 \) indicates that the cell contains a fluid-fluid interface. The local averaged dynamic viscosity \( (\mu) \) is calculated by harmonic averaging (Prosperetti, 2002):

\[
\frac{\rho}{\mu} = F \frac{\rho_1}{\mu_1} + (1 - F) \frac{\rho_2}{\mu_2}
\]
Advection of $F$ depends on the local fluid velocity and is governed by the following conservation equation:

$$\frac{DF}{Dt} = \frac{\partial F}{\partial t} + u \cdot \nabla F = 0$$  \hspace{1cm} (5)

The volumetric surface tension force ($F_\sigma$) is computed using the continuum surface force (CSF) model (Brackbill et al., 1992):

$$F_\sigma = \sigma \kappa n$$  \hspace{1cm} (6)

where $\sigma$ is the coefficient of surface tension, $\kappa$ is local interface curvature and $n$ is the interface normal. $F_\sigma$ is non-zero only at the interface. $n$ and $\kappa$ are first and second order derivative of $F$ respectively as follow:

$$n = \nabla F$$  \hspace{1cm} (7)

$$\kappa = -\nabla \cdot \frac{\nabla F}{|\nabla F|} = -\nabla \cdot \frac{n}{|n|} = \frac{1}{|n|} \left[ \frac{n}{|n|} \cdot \nabla |n| - \left( \nabla \cdot n \right) \right]$$  \hspace{1cm} (8)

2.2. Discretization and solution methodology

The Finite Volume method is used for the discretization of mass and momentum conservation equations on a staggered Cartesian computational grid. In the momentum equation a first-order backward scheme is used for the temporal discretization. A second-order central difference scheme is used for the spatial discretization of diffusion term. A deferred correction has been used for the spatial discretization of advection term which combines the first-order upwind and total variation diminishing min-mod scheme. Both diffusion and advection terms are treated implicitly in the present discretization. Fractional step method is used for the pressure-velocity coupling to compute the divergence-free velocity field. An OpenMP-parallelized Block-Incomplete Cholesky Conjugate Gradient (B-ICCG) solver is used to solve the sparse linear system of equations. Readers are referred to Das et al. (2016) for the intrinsic details on aforementioned discretization and solution methodology. We will only concentrate on the discretization of surface tension force ($F_\sigma$) using the balanced-force concept in the subsequent section.

In the present VOF method piecewise linear interface calculation (PLIC) (Youngs, 1982) is used to represent the fluid-fluid interface by a line in 2D (a plane in 3D). Volume fraction ($F$) and interface normal ($n$) is stored at the cell center of the individual computational cell. $n$ is first evaluated at the four vertices of the computational cell in 2D (eight in 3D) using a central-difference discretization of Eq. (7) and then averaged at the cell center (Parker and Youngs, 1992). The interface position and orientation can be uniquely defined by the combination of $F$ and $n$ (Scardovelli and Zaleski, 2000). Advection of $F$, governed by Eq. (5), is solved explicitly by the pseudo-Lagrangian geometrical advection schemes to minimize numerical diffusion. Readers are referred to Van Sint Annaland et al. (2005) for the necessary mathematical and geometrical details for the treatment of VOF advection.

2.3. Balanced-force concept

Spurious velocities are an undesired by-product of all interface tracking/capturing methods including VOF (Harvie et al., 2006) and arises due to two reasons: i) improper discretization of $F_\sigma$ and ii) inaccuracy in the curvature computation. Here, we will focus on the first problem. Due to the improper discretization of $F_\sigma$ there might be an imbalance of pressure and surface tension forces at the interface which would act as a source of spurious velocities. The balanced-force concept (Francois et al., 2006) eliminates this problem by discretizing the pressure and surface tension forces at the same computational location. In the staggered grid arrangement the pressure force (arising due to $\nabla p$) is discretized at the face centers and hence $F_\sigma$ also
needs to be discretized at the same location as shown in Fig. 1. So, Eq. (6) can be discretized on the face center in x direction as follows:

\[
F_{\sigma x,(i+1/2,j)} = \sigma \kappa_{(i+1/2,j)} \left( \frac{F_{(i+1,j)} - F_{(i,j)}}{\Delta x} \right)
\]

(9)

where \( F_{\sigma x} \) is the surface tension force in x direction. Note that \( F_{\sigma x} \) will only be non-zero in the vicinity of the interface where the gradient term, \( \frac{F_{(i+1,j)} - F_{(i,j)}}{\Delta x} \), is non-zero. A similar formulation can also be adopted for the surface tension force in y direction (\( F_{\sigma y} \)). \( \kappa_{(i+1/2,j)} \) in Eq. (9) is the face-centered curvature. However, all the curvature finding techniques detailed in section 3 focus on computing cell-centered curvatures. Hence, the following weighted interpolation technique is used for interpolating the cell-centered curvatures to face centers:

\[
\kappa_{(i+1/2,j)} = \frac{w_{(i,j)} \kappa_{(i,j)} + w_{(i+1,j)} \kappa_{(i+1,j)}}{w_{(i,j)} + w_{(i+1,j)}}
\]

(10)

\[
w_{(i,j)} = F_{(i,j)}(1 - F_{(i,j)})
\]

(11)

Note that the weight \( w \) tends to 0 as \( F \) tends to 1 or 0. The advantage of using \( F(1 - F) \) as a weighing function will be explained in the subsequent sections.

3. Curvature computation

In this section we will focus on the cell-centered curvature computation from the abruptly-varying volume fraction (\( F \)) field. We will first discuss the convolution and height function method in necessary details and subsequently propose a hybrid CV-GHF method.

3.1. Convolution method

To compute the interface normal (\( n \)) and curvature (\( \kappa \)) in accordance with Eq. (7) and Eq. (8) respectively one needs to perform numerical differentiation of the Heaviside \( F \) field which leads to high frequency aliasing errors. The convolution (CV) method uses a smoothing kernel to convolute the \( F \) field which generates a smoothly varying \( \tilde{F} \) field. \( n \) and \( \kappa \) are then computed by replacing \( F \) with \( \tilde{F} \) in Eq. (7) and Eq. (8). Different smoothing kernels are developed and they can be classified based on two main parameters: i) type of smoothing function and ii) smoothing stencil length \( (h) \). Francois et al. (2006) and Cummins et al. (2005) use sixth and eighth order polynomials as a smoothing kernel, respectively. They have presented the convergence behavior of CV method with different smoothing lengths: i) \( h = \text{constant} \), ii) \( h \propto \sqrt{\Delta} \) and iii) \( h \propto \Delta \). Here, \( \Delta \) denotes the grid spacing. Both reported similar findings which suggests that \( h = \text{constant} \) shows second-order convergence with relatively high errors at the lower grid resolutions. \( h \propto \sqrt{\Delta} \) shows first-order convergence. \( h \propto \Delta \) shows positive convergence at the lower grid resolutions when the interface is not adequately resolved by the grid but, once resolved, shows divergence at higher grid resolutions. So overall, the CV method doesn’t converge for \( h \propto \Delta \) due to the fixed size of discretization stencil. Note that, \( h = \text{constant} \) and \( h \propto \sqrt{\Delta} \) maintains the convergence by increasing the size of the smoothing stencil which might be computationally prohibitive at higher grid resolutions.

In the present work, the smoothing process is performed using the following equation:

\[
D(x) = \begin{cases} 
\frac{1}{2} \left( 1 - \frac{|x|}{h} \right), & |x| \leq h \\
0, & |x| > h
\end{cases}
\]

(12)
\[
\bar{F}(x, y, z) = \sum_mD(x-x_m)D(y-y_m)D(z-z_m) F(x_m, y_m, z_m)
\]

(13)

where the smoothing length of \( h = 2\Delta \) is used. So, the smoothing process would effectively be restricted within a compact \( 3 \times 3 \times 3 \) stencil in 3D. Due to this advantage of locality the present CV method is quite robust as to produce stable interfaces even at extremely low grid resolutions in 3D (see section 5.2 for details). As reported by other researchers [Francois et al. 2006; Cummins et al. 2005] and observed by us, due to the fixed smoothing stencil the present CV method doesn’t converge. However, we will show in section 5.1 that the hybrid CV-GHF method does converge. The reason for this convergence behavior will also be discussed.

3.2. Height function method

In the Height Function (HF) method heights are computed by summing the volume fractions \( F \) across the interface. The preferred direction of summation is the direction of the largest component of interface normal \( n \). This produces a differentiable set of heights from an abruptly-varying \( F \) field. Curvature is then computed from heights by a standard second-order finite difference scheme. The HF method can be classified in two ways according to the type of summation stencil used to compute heights: i) fixed stencil [Francois et al. 2006; Cummins et al. 2005] and ii) adaptive stencil [Popinet 2009; Sussman and Ohta, 2007]. Francois et al. (2006) illustrates the performance of the HF method using fixed \( 3 \times 3, 5 \times 3 \) and \( 7 \times 3 \) stencils in 2D and concludes that \( 7 \times 3 \) stencil is essential to compute accurate curvatures. In the present work the HF method with a fixed \( 7 \times 3 \) stencil in 2D (\( 7 \times 3 \times 3 \) in 3D) is referred as the Standard Height Function (SHF) method. The Generalized Height Function (GHF) method by Popinet (2009) uses an adaptive stencil which changes its size depending on the local interface topology. A 2D implementation of both methods is explained in subsequent sections using Fig. 2.

3.2.1. Standard height function method

Consider a cell \((i, j)\) on the interface \((0 < F < 1)\) where we seek to compute heights using the SHF method. To do so, a fixed \( 7 \times 3 \) stencil is constructed around the cell \((i, j)\) in the direction of the largest component of \( n \). This stencil would include 3 cells above and 3 cells below the reference cell \((i, j)\) in each column. For the cell \((i, j)\) in Fig. 2, \( |n_y| > |n_x| \) and hence the stencil is constructed in \( y \) direction. Height \( H_{i'} \) is then computed in each individual column \( (i') \) by summing \( F \) using the following expression:

\[
H_{i'} = \sum_{j'=j-3}^{j'=j+3} F(i', j'\Delta y) \text{ for } i' = i - 1, i, i + 1
\]

(14)

One important thing to note is that the curvature is computed only if the interface position in the central column derived from the height is located in cell \((i, j)\). This condition eliminates a lot of interfacial cells (especially in 3D) from the curvature computation. For the case (a) of Fig. 2, height in the central column does belong to cell \((i, j)\) and for case (b) it doesn’t. So, the curvature will only be computed for case (a) whereas for case (b) it would be non-existent (numerically zero).

3.2.2. Generalized height function method

Consider the same cell \((i, j)\) on the interface \((0 < F < 1)\) where we now seek to compute heights using the GHF method. In the GHF method, we first try to build an adaptive stencil in the direction of the largest component of \( n \). This procedure is followed column-wise and involves a central and all neighboring columns (total 3 in 2D or 9 in 3D). A column is called consistent if it has a base \((F = 1)\) and a top \((F = 0)\) with
a strictly monotonic variation of \( F \) between the base and top. If all columns in this direction are consistent then the height \( (H_{i'}) \) is computed by summing \( F \) from the base to top in each individual column \( (i') \) using the following expression:

\[
H_{i'} = \sum_{j'=\text{base}}^{j'=\text{top}} F(i', j') \Delta y \quad \text{for} \quad i' = i - 1, i, i + 1
\]

The computed heights need to be normalized relative to the origin (base of the central column) of the adaptive stencil using:

\[
H_{i'} = H_{i'} - \text{sign}(n_y)(\text{base}_{i'} - \text{origin}) \Delta y \quad \text{for} \quad i' = i - 1, i, i + 1
\]

If at least one column is found to be inconsistent while trying to build an adaptive stencil in the particular direction then the same procedure is repeated for the alternative directions in the decreasing order of the componental magnitude of \( \mathbf{n} \). The ultimate goal is to search the direction for which all columns are consistent. For the cell \((i, j)\) in Fig. 2, \(|n_y| > |n_x|\) and hence we first try to build an adaptive stencil in \( y \) direction. For the cases presented in Fig. 2 (a) and (b), all columns in \( y \) directions are consistent and hence there is no need to try for the alternative direction. But if, by chance, at least one column in \( y \) direction would have been found to be inconsistent then we would have tried the alternative \((x)\) direction to find all consistent columns. This type of consistency check is not embodied in the SHF method and hence GHF method computes substantially more accurate heights especially for highly curved interfaces. See section 4.2 for more details on this subject.

If it is impossible to find all consistent columns in any direction then the GHF method for cell \((i, j)\) is termed inconsistent and won’t be used for computing the curvature. This would generally happen at low grid resolutions when the interface is not adequately resolved on the Eulerian grid.

For the consistent GHF formulation, curvatures are computed irrespective of the location of the height (within or outside cell \((i, j)\)) in the central column. So, for both cases (a) and (b) of Fig. 2 curvature will be computed. This subtle difference in implementation (compared to SHF method) drastically improves the performance of GHF method by eliminating the spurious velocities for the stationary interfaces (Popinet, 2009). A detailed discussion on this topic is presented in section 4.3.

### 3.2.3. Height function curvature

Once heights are obtained from either SHF or GHF method, curvature is computed using following expression:

\[
\kappa = \begin{cases} 
- \frac{H_{xx}}{(1 + H_x^2)^{\frac{3}{2}}}, & \text{in 2D} \\
- \frac{H_{xx} + H_{yy} + H_{zx}H_y^2 + H_{zy}H_z^2 - 2H_{xy}H_xH_z}{(1 + H_x^2 + H_y^2)^{\frac{3}{2}}}, & \text{in 3D}
\end{cases}
\]

Note that in 2D Eq. (17) is valid for the case where a HF stencil is constructed in the \( y \) direction as shown in Fig. 2. Similarly, in 3D Eq. (17) would be used when a HF stencil is constructed in the \( z \) direction for a 3D curvature computation. Eq. (17) can be extended straightforwardly for other directions and all the derivatives \((H_x, H_y, H_{xx}, H_{yy} \text{ or } H_{xy})\) are computed using central finite differences. The HF curvature is second-order accurate and a formal proof of the same is presented by Bornia et al. (2011).
3.3. Hybrid CV-GHF method

In this section, a novel hybrid CV-GHF method is presented combining the convolution (CV) and the generalized height function (GHF) method. As reported by Cummins et al. (2005) and observed as well by us, the CV method exhibits a high degree of robustness compared to the HF method when the interface is not adequately resolved on the grid. This typically happens at lower grid resolutions. However, the HF method is quite accurate and exhibits second-order convergence at higher grid resolutions. So, we propose to use a hybrid approach which harnesses the robustness of the CV method at regions where the interface is poorly resolved and computes accurate curvature using the HF method at regions where the interface is well resolved.

A branching algorithm is an integral part of the the hybrid method that decides which method (CV or HF) is to be used depending on the local interface topology. The reason for choosing GHF over SHF method is due to two reasons: i) GHF has an in-built branching algorithm as a consistency check and ii) GHF method computes substantially more accurate heights (and in-turn curvature) than the SHF method (section 4).

A straightforward procedure to implement hybrid CV-GHF method is presented henceforth. Consider a cell \((i,j)\) on the interface \((0 < F < 1)\) where curvature is to be computed using CV-GHF method. The first step is to check if a consistent GHF formulation is possible for the cell \((i,j)\). If the outcome is positive, compute the curvature using GHF method. If the outcome is negative, use CV method instead. This way, the present hybrid approach automatically switches the curvature computation method from GHF to CV at lower grid resolutions.

4. Comparison of standard and generalized height function methods

In this section, we will present different cases for which the generalized height function (GHF) method computes substantially more accurate heights (and in-turn curvature) compared to standard height function (SHF) method. Direct comparison between GHF and SHF would affirm our choice of using GHF in the present hybrid CV-GHF method.

4.1. Nearby interfaces

In the numerical simulations of multiphase flows it is quite normal to have multiple fluid-fluid interfaces nearby each other. A practical example would be merging or breakup of interfaces. Fig. 3 presents a 2D case to illustrate such scenario using circular interfaces. It also includes heights and curvatures computed using SHF and GHF methods for the reference cell. Note that the summation stencil to compute heights is constructed in \(y\) direction as \(|n_y| > |n_x|\) for the present case. It is clear that the SHF method computes quite an inaccurate curvature due to the presence of multiple interfaces in the large \(7 \times 3\) stencil. It has been observed that this problem with SHF method typically arises when interfaces are less than 4 grid cells apart. GHF method eliminates this problem by constructing an adaptive stencil which is quite compact for the present case. It would exclude the presence of nearby interfaces in the heights computation and hence computes very accurate curvature.

4.2. Direction of height function stencil

SHF method constructs a fixed \(7 \times 3\) stencil strictly in the direction of the largest component of \(n\). However, GFH is more flexible on this particular aspect. It would first try to construct an adaptive stencil in the direction of the largest component of \(n\) (same as SHF). But if at least one column is inconsistent in that direction then it also tries alternative directions in the decreasing order of the componental magnitude
of \( n \) until a direction with all consistent columns is found. This flexibility helps GHF method in computing more accurate curvatures especially for highly curved interfaces.

Fig. 4 presents a 2D case to demonstrate such scenario using an elliptical interface. It also includes heights and curvatures computed using the SHF and GHF methods for the reference cell. For this cell, \( |n_x| > |n_y| \) and hence the SHF method constructs a fixed 7 \& 3 stencil in \( x \) direction to compute heights. Note that the highly curved elliptical interface passes through the SHF stencil two times. Due to this, a poor estimation of heights (and hence curvature) is obtained compared to the exact curvature \( \kappa_{\text{exact}} \), which is analytically calculated for the nearest point on the elliptical interface from the cell center of the reference cell. GHF would also start with the \( x \) direction and tries to construct an adaptive stencil for each individual column. However, one of the columns in \( x \) direction is inconsistent because it doesn’t show monotonic variation of \( F \) across the interface. Hence, the GHF method is inconsistent in the \( x \) direction. However, the GHF method doesn’t stop here and tries to find all consistent columns in an alternative \( y \) direction. For the present case, the GHF method successfully constructs an adaptive stencil with all consistent columns in \( y \) direction. Heights and curvature computed using such consistent GHF formulation are by far more accurate than the curvature estimated by the SHF method.

### 4.3. Curvature convergence behavior

In section 3.2 a condition is mentioned which suggests to compute the HF curvature only if the height in the central column (of the HF stencil) belongs to the cell where the curvature is to be computed. This condition will be refereed as ‘HF condition’ for further discussion. SHF (in its original form) follows this condition and hence eliminates a lot of interfacial cells \((0 < F < 1)\) from the curvature computation. However, GHF doesn’t follow this condition and computes curvature for all the cells on the interface. In this section, we will focus on the convergence of the HF curvature for the different cases described in Table I. It is important to note that Francois et al. (2006); Cummins et al. (2005) use SHF with HF condition (Case 1) as explained in section 3.2.1. A 3D spherical interface is considered as a model test problem for this purpose. A sphere of radius \( R \) is initiated with different grid resolutions \((R/\Delta)\) and curvature error norms are defined as follows:

\[
\begin{align*}
L_2(\kappa) &= \frac{1}{\kappa_{\text{exact}}} \sqrt{\sum_i (\kappa_i - \kappa_{\text{exact}})^2} \\
L_\infty(\kappa) &= \frac{1}{\kappa_{\text{exact}}} \max_i (|\kappa_i - \kappa_{\text{exact}}|)
\end{align*}
\]

where \( i \) is the interfacial cell index which considers cells in accordance to the HF condition (being followed or not) for different cases listed in Table I. \( \kappa_{\text{exact}} \) is the analytical curvature which is equal to \( 2/R \) for the spherical interface. Liovic et al. (2010) uses a multiplying factor of \( F(1-F) \) to define the \( L_2 \) error norm. So, to make the present discussion more enriching, we have also defined \( L_2 \) error norm similarly:

\[
L_{2, F(1-F)}(\kappa) = \frac{1}{\kappa_{\text{exact}}} \sqrt{\sum_i F_i(1-F_i)(\kappa_i - \kappa_{\text{exact}})^2} \\
\sum_i F_i(1-F_i)
\]

Fig. 5 shows the convergence of \( L_2 \) and \( L_\infty \) error norms for the cases listed in Table I. We have considered grid resolutions \((R/\Delta)\) of 6, 12, 24 and 48, same as Liovic et al. (2010) for this study. However, using different grid resolutions would not change the presented outcome. As reported by other researchers, HF curvature obtained from the heights computed using SHF method converges with second-order (Case 1). However, it
does so only while following the HF condition. If we don’t follow the HF condition and use SHF method to compute heights for all the cells on the interface (Case 2) then the curvature doesn’t converge. Now, in the same scenario if we use GHF method to find heights (Case 3) then the curvature retains second-order convergence when GHF method is consistent for all the interfacial cells ($R/\Delta > 16$). Note that the GHF method is not consistent for all interfacial cells at lower grid resolutions ($R/\Delta < 16$) and hence error norms have been calculated considering those interfacial cells where GHF is consistent.

The local curvature error ($\delta \kappa_i$) for an individual cell ($i$) is defined as follows:

$$\delta \kappa_i = \frac{1}{\kappa_{exact}}(|\kappa_i - \kappa_{exact}|)$$ (21)

Fig. 6 (a), (b) and (c) show the contours of $\delta \kappa_i$ with the azimuth angle ($\theta$) and polar angle ($\phi$) indicating the (projected) positions of the interfacial cells on the sphere for the case 1,2 and 3, respectively. A spherical interface with $R/\Delta = 24$ is considered here. Total number of interface cells ($0 < F < 1$) for this case is 10712. It is evident that the point density in case 1 is less than that of case 2 or 3. The reason for this is that the case 1 follows HF condition and hence eliminates $4664$ ($\approx 44\%$) interfacial cells from the curvature computation. $\delta \kappa_i$ shows a similar pattern for all the cases. It is highest for the cells where $|n_x| \approx |n_y| \approx |n_z|$ as the interface normal is oriented away from the (Cartesian) direction of HF stencil (López et al., 2009). For a spherical interface this condition would arise in the diagonal directions ($|x| \approx |y| \approx |z|$). Interestingly, SHF in case 2 computes very inaccurate curvature near the diagonal directions and the errors don’t converge with increased grid resolution because a limited $7 \times 3$ stencil is not sufficient to capture the full interface topology in this region. However, GHF in case 3 computes quite accurate curvature (on the same interfacial cells as case 2) using a large adaptive stencil. The computed curvature in case 3 also shows second-order convergence.

Case 4 uses the SHF method for heights and computes curvature for all the interfacial points (same as case 2). However, the only difference compared to case 2 is the definition of $L_2$ error norm. Case 4 uses a multiplying factor $F(1 - F)$ to define the $L_2$ error norm (Eq. (20)) and case 2 doesn’t (Eq. (18)). Due to this subtle difference case 4 maintains the second-order convergence in curvature error. However, curvature errors in case 2 don’t converge. This suggests that the inaccuracy in curvature computation is prone to increase for the cells where $F$ is close to 0 or 1. Hence, it would be more logical to use $F(1 - F)$ as a weighing factor instead of the simple arithmetic averaging used by François et al. (2006); Popinet (2009) to interpolate the computed cell-centered curvatures to face centers in the balanced-forced concept (section 2.3).

5. Results

In this section a variety of 3D test cases will be presented to verify/validate the present hybrid CV-GHF method. First, a curvature convergence study is presented using 100 randomly initiated droplets. Subsequently, a classical test case of a stationary droplet is presented to check the accuracy of the present method with respect to various parameters i.e. magnitude of spurious velocity, Laplace pressure difference across the interface and droplet shape distortion. Next, a test case of droplet oscillation is presented and the obtained results are compared with the analytical solution of Lamb (1932). Lastly, a test case of buoyant bubble rise in a quiescent liquid is presented. Simulations are performed for a wide range of Eötvös and Morton numbers to compare the terminal rise velocity with experimental (Grace, 1973) and numerical (Baltussen et al. 2014) results. Readers are referred to Das et al. (2016) for the validation/verification of the Navier-Stokes solver and Baltussen et al. (2014) for the mass conservation in VOF advection.
For all simulations the time step ($\Delta t$) has been chosen such that it satisfies both Courant–Friedrichs–Lewy (CFL) and capillary criteria as follows:

$$\Delta t < \min (\Delta t_{\text{CFL}}, \Delta t_{\sigma}) = \min \left( \frac{\Delta}{U_{\text{max}}} \sqrt{\frac{(\rho_1 + \rho_2)\Delta^2}{4\pi \sigma}} \right)$$  \hspace{1cm} (22)

Here, $\Delta t_{\text{CFL}}$ and $\Delta t_{\sigma}$ are the minimum required time steps according to the CFL and capillary criteria, respectively. $\Delta$ is the computational grid size and $U_{\text{max}}$ is the maximum fluid velocity in the computational domain.

5.1. Convergence of CV-GHF curvature

In this section, a grid convergence study is presented to check the accuracy of the curvature computed using the present hybrid CV-GHF method. This test is carried out using 100 randomly placed droplets with a fixed radius ($R$). The location of the center of the droplet ($x_0, y_0, z_0$) varies in the region of $\pm \Delta/2, 2R \pm \Delta/2, 2R \pm \Delta/2$ in the computational domain of $4R \times 4R \times 4R$. The volume fraction ($F$) field is initiated using a second-order accurate method explained in the appendix. The random initiation takes all the different relative alignments of grid and interface into account. The grid convergence study has been performed for a wide range of grid resolutions ($R/\Delta$ from 2 to 64) and the curvature error norms are computed as follows:

$$L_2(\kappa) = \frac{1}{\kappa_{\text{exact}}} \sqrt{\sum_n \sum_i (\kappa_{n,i} - \kappa_{\text{exact}})^2}$$  \hspace{1cm} (23)

$$L_\infty(\kappa) = \frac{1}{\kappa_{\text{exact}}} \max_n (|\kappa_{n,i} - \kappa_{\text{exact}}|)$$  \hspace{1cm} (24)

where indexes $n$ and $i$ denote $i^{th}$ interfacial cell ($0 < F < 1$) on $n^{th}$ droplet. Fig. 7 shows $L_2$ and $L_\infty$ curvature error norms for the CV-GHF method. It also shows the fraction of interfacial cells using the CV method (as GHF method is inconsistent) for the curvature computation. At extremely low grid resolutions ($R/\Delta \leq 2$), all of the interfacial cells use the CV method. As grid resolution is increased ($R/\Delta$ between 2 to 16), more cells can have consistent GHF formulation but CV method will still be actively used. At higher resolutions ($R/\Delta \geq 16$), all of the interfacial cells use the GHF method. The $L_\infty$ error norm shows first-order convergence for $R/\Delta < 16$ and is mostly governed by CV method. However, second-order convergence is observed for $R/\Delta > 16$ due to the GHF method. $L_2$ error norms show a smooth transition from first-order at lower ($R/\Delta < 4$) to second-order at higher ($R/\Delta > 16$) grid resolutions.

Balanced-force concept needs face-centered curvatures to apply the surface tension force on the staggered computational grid as explained in section 2.3. The present method uses a weighing factor of $F(1 - F)$ to interpolate the computed cell-centered curvatures to face centers. To measure its effectiveness, $L_2$ and $L_\infty$ error norms are also computed for the face-centered curvature using the same Eq. (23) and (24), with index $i$ denoting the face center. Compared to the cell-centered $L_2$ curvature error norm, the face-centered one is 17.57%, 9.87%, 5.88% and 4.73% more accurate in the cases of $R/\Delta = 2, 4, 8$ and 16, respectively. As the grid resolution increases, this improvement becomes less significant. Unfortunately, the factor $F(1 - F)$ couldn’t reduce the face-centered $L_\infty$ error norm as the maximum curvature error occurs for the cells where $F$ is near to 0 or 1. For such cells, face-centered curvature using Eq. (10) will only include that cell ($F$ is near to 0 or 1) as the curvature value for the neighboring cell ($F = 0$ or 1, respectively) doesn’t exist.

It is interesting to note that the present CV method uses a fixed length ($h \propto \Delta$) smoothing kernel. Curvatures from such a CV method don’t convergence (Cummins et al., 2005). We have also observed the
same for the present CV method (if used alone). However, in conjunction with GHF the hybrid CV-GHF method shows convergence. The possible reason for this behavior is the nature of the local curvature error \( (\delta \kappa_i) \) for the CV method. Fig. 8 shows the contours of \( \delta \kappa_i \) with \( \theta \) and \( \phi \) for the present CV method for the same test case presented in section 4.3. Curvatures computed from the CV method are most inaccurate in the region where \( \theta \) or \( \phi \) are close to \( 0^\circ \) or \( 90^\circ \). It complements the GHF curvature (Fig. 6) which is most inaccurate in the diagonal directions (\( \approx 45^\circ \)). We believe due to this complementary behavior the hybrid CV-GHF method is more accurate compared to the individual ones.

5.2. Stationary droplet

In this section, we will present a classical test case of a stationary droplet to check the accuracy and robustness of the present CV-GHF hybrid method for static interfaces. Ideally, for a stationary droplet pressure and surface tension forces balances each other producing a Laplace pressure difference across the droplet interface. But exact force balance is difficult to obtain in the numerical simulations which leads to undesired spurious velocities. Using GHF method in conjunction with balanced-force discretization, Popinet (2009) obtained the exact numerical balance of pressure and surface tension forces (within round-off errors) for static interfaces. However, GHF method is applicable only at the higher resolutions as discussed before.

A spherical droplet of radius \( R = 1 \) is placed at the center of \( 4R \times 4R \times 4R \) domain. Boundaries of this domain are of the free-slip type. The physical properties of the droplet (1) and the outside fluid (2) are as follows: density \( \rho = \rho_1 = \rho_2 = 500 \text{ kg/m}^3 \), viscosity \( \mu = \mu_1 = \mu_2 = 0.1 \text{ Pa·s} \) and surface tension \( \sigma = 0.01 \text{ N/m} \). With these properties, the Laplace number \( (La = 2R\sigma\rho/\mu^2) \) is 1000. Simulations are performed for a wide range of grid resolutions with \( R/\Delta \) ranging from (as low as) 2 to 64. Interestingly, we could obtain a stable droplet interface even at extremely low grid resolution of \( R/\Delta = 2 \). We attribute this to the robustness of the present CV method which uses a quite compact stencil for the curvature computation.

First, we focus on quantification of the spurious velocities for the present hybrid method. RMS \( (L_2) \) and maximum \( (L_\infty) \) spurious velocity norms are defined as follows:

\[
L_2(V) = \sqrt{\frac{\sum_i |V_i|^2}{\sum_i}} \\
L_\infty(V) = \max_i(|V_i|)
\]

where \( |V_i| \) is the magnitude of the dimensionless spurious velocity of the \( i^{th} \) grid cell. Note that the non-dimensionalization is performed by \( \sqrt{\sigma/\rho D} \) which denotes the scale of velocities associated with a capillary wave of wavelength comparable to the diameter of the droplet \( (D) \). Fig. 9(a) shows the transient evolution of the maximum spurious velocity as a function of dimensionless time \( (t/T_v) \). Here, \( T_v = \rho D^2/\mu \) is the viscous time-scale of the droplet. Popinet (2009) suggests that the stationary droplet simulations should at least be run up to the time comparable to \( T_v \) to check the accuracy of the surface-tension implementation. This provides sufficient time for the droplet to attain equilibrium (if it can) by slowly dissipating the kinetic energy. Most of the literature (François et al., 2006; Liovic et al., 2010; López et al., 2009) doesn’t adopt this criterion and reports the spurious velocity magnitude only for the initial time steps. For the present test case, simulations are run up to \( t/T_v = 1 \) and Fig. 9(a) shows it is sufficient for the droplet to stabilize. Table 2 reports \( L_2 \) and \( L_\infty \) spurious velocity norms at \( t/T_v = 1 \). It’s clear that the spurious velocities decrease with increasing grid resolution for \( R/\Delta < 16 \). This is due to the gradual switch from the CV to the GHF method for the curvature computation as shown in Fig. 7. For \( R/\Delta \geq 16 \), spurious velocities becomes zero (up to
round-off errors) as for all of the interfacial cells the GHF method is used. We have considered numerical values less than $10^{-10}$ to be essentially zero due to round-off errors.

Next, we compute $L_2$ and $L_\infty$ curvature error norms using Eq. (18) and (19), respectively at $t/T_v = 1$. Fig. 9(b) shows the convergence of curvature error norms with grid resolution. The behavior is similar to the one observed in Fig. 7. However, the absolute value of the error norms are lower. Interestingly, in presence of surface tension force, the $F$ field gradually rearranges itself from the initial exact distribution ($F_{\text{exact}}$) to minimize the curvature standard deviation ($\text{std}(\kappa)$) and in-turn reducing the curvature error norms. $\text{std}(\kappa)$ is defined as follows:

$$\text{std}(\kappa) = \frac{1}{\kappa_{\text{exact}}} \sqrt{\frac{\sum_i (\kappa_i - \langle \kappa \rangle)^2}{\sum_i}}$$  \hspace{1cm} (27)

where $\kappa_i$ is the computed curvature for the $i^{th}$ interface cell whereas $\langle \kappa \rangle$ is the mean curvature of all interface cells. The temporal evolution of $\text{std}(\kappa)$ for different grid resolutions follows a similar trend as spurious velocities in Fig. 9(a). Table 2 reports $\text{std}(\kappa)$ for different grid resolutions at $t/T_v = 1$. As grid resolution increases $\text{std}(\kappa)$ reduces because more and more interface cells use the highly accurate GHF method for curvature computation. For $R/\Delta \geq 16$, $\text{std}(\kappa)$ becomes zero (up to round-off errors) as all of the interface cells use the GHF method. This can also be verified by an equal $L_2$ and $L_\infty$ curvature error norms in Fig. 9(b) for $R/\Delta \geq 16$. As $\text{std}(\kappa)$ approaches zero, all interfacial cells possess the same curvature and hence same surface tension force. In this case surface tension and pressure forces can balance each other exactly and the droplet is in equilibrium. So, even if the computed curvature is not exact (Fig. 9(b) presents non-zero curvature error norms), spurious velocities becomes zero as $\text{std}(\kappa)$ is zero. Hence, it is more precise to state that the source of the spurious velocities is spatial fluctuations of the computed curvature rather than the curvature errors itself.

A stationary droplet should ideally experience a Laplace pressure difference ($\Delta p_{\text{exact}} = 2\sigma/R$) across the interface. For the present simulations, the maximum Laplace pressure difference ($\Delta p_{\text{max}}$) and pressure error $E(\Delta p)$ are defined as follows:

$$\Delta p_{\text{max}} = \max_i (p_i) - \min_i (p_i)$$  \hspace{1cm} (28)

$$E(\Delta p) = \frac{1}{\Delta p_{\text{exact}}} |\Delta p_{\text{max}} - \Delta p_{\text{exact}}|$$  \hspace{1cm} (29)

where $p_i$ is the pressure of the $i^{th}$ grid cell. Fig. 9(c) shows the convergence of $E(\Delta p)$ with grid resolution at $t/T_v = 1$. The convergence behavior is same as exhibited by curvature as shown in Fig. 9(b). Interestingly, at higher grid resolutions ($R/\Delta \geq 16$) $L_2$ and $L_\infty$ curvature error norms become exactly equal to the pressure error. This shows that the error in the curvature computation is exactly translated into the pressure error.

As discussed before, in presence of a surface tension force $F$ field rearranges itself from the initial exact distribution ($F_{\text{exact}}$) which suggests that the equilibrium droplet shape at $t/T_v = 1$ is not exactly equal to the initial spherical shape. So, it is important to compute the droplet shape errors. $L_2$ and $L_\infty$ shape error norms are defined as follows:
\[ L_2(\text{shape}) = \sqrt{\sum_i (F_i - F_{i,\text{exact}})^2} \]  
(30)

\[ L_\infty(\text{shape}) = \max_{i}(|F_i - F_{i,\text{exact}}|) \]  
(31)

where \( F_{i,\text{exact}} \) and \( F_i \) represent the volume fraction of the \( i^{th} \) interface cell at \( t/T_v = 0 \) and 1, respectively.

Fig. 9 (d) shows the convergence of \( L_2 \) and \( L_\infty \) shape error norms with grid resolution. Again, it follows the same trend as Fig. 9 (b) or (c) at lower grid resolutions. However, at higher grid resolutions \( (R/\Delta \geq 16) \) less than second-order convergence is observed.

It is important to note that we have presented the characteristics of the present CV-GHF hybrid method for a stationary droplet in 3D covering the whole range of grid resolutions \( (R/\Delta \text{ from 2 to 64}) \). This includes all possible cases for the curvature computation i.e. CV alone \( (R/\Delta \leq 2) \), both CV and GHF \( (2 < R/\Delta < 16) \) and GHF alone \( (R/\Delta \geq 16) \). P opinet (2009) reports similar results for his hybrid LS-GHF method considering a 2D circular droplet. However, the presented results cover only higher grid resolutions where GHF method alone is used for the curvature computation.

### 5.3. Oscillating droplet

In this section, we consider a test case of an oscillating droplet with no external force to test the present CV-GHF hybrid method for dynamic interfaces. Initially, the viscous droplet is given a small perturbation from its equilibrium (spherical) shape and hence it will experience a damped harmonic oscillation. The shape of the droplet is initialized according to:

\[ R = R_0[1 + \eta P_2(\cos \phi)] \]  
(32)

where \( R_0 \) is the equilibrium radius, \( \phi \) is the polar angle, \( \eta \) is the perturbation amplitude and \( P_2 \) is the Legendre polynomial of second order. The droplet is initially placed at the center of \( 4R_0 \times 4R_0 \times 4R_0 \) domain. Simulations are performed with \( R_0 = 1 \) m and \( \eta = 0.04 \) m with all free-slip boundaries. The physical properties of the droplet (1) and the outside fluid (2) are as follows: density \( \rho_1 = 1, \rho_2 = 0.01 \text{ kg/m}^3 \); viscosity \( \mu_1 = 0.01, \mu_2 = 0.0001 \text{ Pa} \cdot \text{s} \) and surface tension \( \sigma = 1 \text{ N/m} \). Physical properties are the same as taken by López et al. (2009) and Liovic et al. (2010) for the identical problem. Lamb (1932) presents an analytical solution of this problem given by:

\[ r(t) = \eta \exp(-t/\tau) \cos(wt) \]  
(33)

\[ w = \sqrt{\frac{24\sigma}{R_0^3[3\rho_1 + 2\rho_2]}} \]  
(34)

\[ \tau = \frac{\rho_1 R_0^2}{5\mu_1} \]  
(35)

where \( r(t) \) is the location of the oscillating interface (with respect to its equilibrium position) at \( \phi = 0^\circ \) and time \( t \), \( w \) is the angular frequency of oscillation and \( \tau \) is the relaxation time-scale.

Simulations are performed with different grid resolutions i.e. \( R_0/\Delta = 8, 16 \) and 32. Fig. 10 shows the dimensionless location of the interface \( (r(t)/R_0) \) with dimensionless time \( (t/\tau) \). An excellent qualitative match is observed compared to the analytical solution given by Eq. (33). For a quantitative comparison,
time period of the oscillation \( T = 2\pi/w \) given by Eq. (34) is compared with the numerical value \( (T^*) \) and the percentage error \( \left| (T - T^*) \times 100/T \right| \) is presented in Table 3. It also shows the \( L_2 \) and \( L_\infty \) interface location error norms defined as follows:

\[
L_2(r) = \frac{1}{\eta} \sqrt{\frac{\sum_i (r(t_i) - r^*(t_i))^2}{\sum_i}} \quad (36)
\]

\[
L_\infty(r) = \frac{1}{\eta} \max_i |(r(t_i) - r^*(t_i))| \quad (37)
\]

where \( i \) is the index to denote time step during the first oscillation period, \( r(t) \) is the analytical location given by Eq. (33) and \( r^*(t) \) is numerical location from the simulations. Table 3 shows a good grid convergence behavior and the obtained results are more accurate compared to those of López et al. (2009) and Liovic et al. (2010).

5.4. Rising Bubble under gravity

In this section, we consider a test case of a rising bubble under the influence of gravity to test the present CV-GHF hybrid method for highly dynamic interfaces. Grace (1973) collected experimental data on freely rising single bubbles and characterized the shape and terminal rise velocity. The bubble will attain different shapes i.e. spherical, ellipsoidal, skirted, wobbling etc. based on Eötvös (\( Eo \)) and Morton (\( Mo \)) numbers. These experimental results are combined into the well-known “Grace diagram” which is often used to validate numerical models for multiphase flows. Baltussen et al. (2014) performed VOF simulations of the same test case using several curvature finding methods i.e. CV, HF and Tensile Force (TF). They characterized different regions in the Grace diagram with respect to the performance of the curvature finding methods. Their results provide a good basis for a ‘one-on-one’ comparison of the hybrid CV-GHF method with the individual curvature finding methods (CV, HF or TF).

An initial spherical bubble (1) is released in a viscous fluid (2) under the influence of gravity. Simulations are performed over a wide range of \( Eo \) and \( Mo \) and the terminal rise velocity is characterized as a Reynolds number \( Re \) in dimensionless form. Additional information for the simulations is provided in Table 4. Definitions of the dimensionless numbers are as follows:

\[
Mo = \frac{g\rho_2^2(\rho_2 - \rho_1)}{\rho_2^2\sigma} \quad (38)
\]

\[
Eo = \frac{g(\rho_2 - \rho_1)D^2}{\sigma} \quad (39)
\]

\[
Re = \frac{\rho_2V_{1,\infty}D}{\mu_2} \quad (40)
\]

where \( V_{1,\infty} \) is the terminal rise velocity of the bubble opposite to the direction of gravitational acceleration \( \alpha \). Boundaries of the computational domain are of the free-slip type. van Sint Annaland et al. (2006) performed simulations for the same test case with different computational domain sizes and showed that the present domain size is sufficient to eliminate the boundary effects on \( V_{1,\infty} \).

Table 4 shows the terminal rise velocity (expressed as \( Re \)) for a wide range of \( Eo \) and \( Mo \). It compares the results obtained using the present hybrid CV-GHF method with the experimental results of Grace (1973) and numerical results of Baltussen et al. (2014) from CV, HF and TF methods. Simulation cases with different
combinations of $E_o$ and $M_o$ are selected directly from the results of Baltussen et al. (2014) for ‘one-on-one’ comparison. They report multiple cases where one of the individual curvature finding methods (CV, HF or TF) fails to deliver a stable bubble interface. Also, their results suggests that the inaccuracy in the computed terminal rise velocity is higher for highly dynamic interfaces (generally at higher $Re$). So, we have selected the present simulation cases such that they span the whole Grace diagram and also include the critical cases reported by Baltussen et al. (2014). Note that, Baltussen et al. (2014) performed simulations with 30 grid cells across the initial spherical bubble diameter. However, the present simulations are performed with only 20 grid cells to include the effect of both CV and GHF method in curvature computation.

It is clear from the obtained results that the hybrid CV-GHF method computes more accurate terminal rise velocities in almost all the cases. The difference is specifically noticeable in cases with highly dynamic interfaces prevailing at higher $Re$. For these cases, it is essential to dynamically switch to the appropriate curvature finding method based on the local interface topology and the hybrid CV-GHF method does just that. Bubble shapes obtained from the present simulations are practically the same as those reported by Baltussen et al. (2014). More importantly, the hybrid CV-GHF method produces stable bubble interfaces for all the cases which demonstrates its robustness for highly curved interfaces. So, a single curvature description (CV-GHF) is more accurate and robust then any of the individual one (CV, HF or TF) for the whole range of Grace diagram.

6. Conclusions

The Volume of fluid (VOF) method uses an abruptly-varying volume fraction field to represent a fluid-fluid interface. A sharp interface poses an immense challenge for the accurate computation of the local curvature. Different curvature finding methods (i.e. convolution (CV), height function (HF), reconstructed distance function (RDF), tensile force (TF)) exists but individually they all lack in either robustness or accuracy.

In this paper, a hybrid (CV-GHF) curvature finding method is presented combining the CV and HF method. For the HF curvature, generalized height function (GHF) approach is chosen over the classical standard height function (SHF) due to various important reasons explained in this paper. A branching algorithm, an inherent part of the GHF method, decides the appropriate method for curvature computation based on the local interface topology. This hybrid approach uses a robust CV method at lower grid resolutions where the interface is not adequately resolved by the grid. However, it automatically switches to the accurate GHF method at higher grid resolutions. The hybrid CV-GHF method is first-order accurate at lower grid resolutions ($R/\Delta \leq 4$) and second-order accurate at higher grid resolutions ($R/\Delta \geq 16$) with a smooth transition in between ($4 < R/\Delta < 16$).

The hybrid CV-GHF method is used in conjunction with the balanced force discretization for the surface tension force. It gives an exact equilibrium between the surface tension and pressure force for the static interfaces at higher grid resolutions ($R/\Delta \geq 16$). This fully eliminates the spurious velocities (up to round-off errors) which is one of the biggest challenge in VOF simulations. The CV-GHF method is validated for several test cases i.e. a stationary droplet, an oscillating droplet and the buoyant rise of gas bubbles. Obtained results shows an excellent agreement with analytical/experimental results available in the literature. Specifically, the rising bubble test case shows the real advantage of the hybrid CV-GHF method in terms of both robustness and accuracy compared to the individual curvature finding methods.
Appendix: Volume fraction initiation

It is quite important to discuss the accuracy of volume fraction \((F)\) field initiation as it dictates the accuracy of the curvature computation. We used local mesh refinement technique to initiate \(F\) field in 3D. Each Cartesian computational cell containing interface \((0 < F < 1)\) is subdivided to the \(l^{th}\) level and interface is reconstructed using piece-wise linear segments (plane in 3D or line in 2D) at the finest level. \(F\) for a cell \((i, j)\) can computed from the following expression:

\[
F_{(i,j)} = \frac{\sum_k \Delta V_k}{\Delta V}
\]

where \(\Delta V_k\) is the volume under the piece-wise linear segment in each subdivision \(k\) and \(\Delta V = \Delta x \Delta y \Delta z\) is the cell volume. Fig. 11 is presented in 2D to understand this procedure more clearly. First thing is to decide if the computational cell contains the interface or not. If all the vertices of the cell are either in fluid 1 \((F = 1)\) or in fluid 2 \((F = 0)\) then it doesn’t contain the interface otherwise it does \((0 < F < 1)\). For a single computational cell \((i, j)\) presented in Fig. 11, 1 vertex \((P)\) is in fluid 1 and 3 vertices \((M, N\) and \(O)\) are in fluid 2 and hence this cell contains the interface. To compute the accurate \(F\) at the cell center of this cell, it is subdivided to the \(l^{th}\) level. For simplicity, Fig. 11 uses \(l = 2\), however in practice we have used \(l = 4\). For each subdivision, we can decide if it contains the interface (or not) using the same method described before (for cell). Subdivision \(a\) and \(b\) have all their vertices in fluid 2 and 1, respectively. So, they don’t contain the interface. However, subdivision \(c\) contains the interface as it has vertices lying in both fluid 1 and 2. For each subdivision (like \(c\)), interface is approximated as a piece-wise linear segment. \(F\) for cell \((i, j)\) can then be computed using Eq. (41) in 2D by summing the area under these segments.

This procedure is quite robust and produces a second-order accurate \(F\) field. For the test case presented in section 5.1, we can define the \(L_2\) volume error norm using 100 randomly initiated droplet as follows:

\[
L_2(V) = \frac{1}{V_{exact}} \sqrt{\sum_n (V_n - V_{exact})^2}
\]

where \(V_n\) is the initiated volume using the present method for \(n^{th}\) droplet and \(V_{exact} = \frac{4}{3} \pi R^3\) is the analytical volume. Fig. 12 shows the convergence of \(L_2\) volume error norm using different grid resolutions. The volume initiation is quite accurate even at lower grid resolutions and converges exactly with second-order.
Acknowledgments

This work is part of the Industrial Partnership Program (IPP) ‘Computational sciences for energy research’ of the Netherlands Organization for Scientific Research (NWO) and co-financed by Shell Global Solutions International B.V. This work was carried out on the Dutch national e-infrastructure with the support of SURF Cooperative.
References


Table 1: Case description to analyze HF curvature convergence behavior.

<table>
<thead>
<tr>
<th>Cases</th>
<th>Method to compute heights</th>
<th>Is HF condition followed while computing curvature?</th>
<th>Curvature error ($L_2$)</th>
<th>Curvature error ($L_\infty$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1</td>
<td>SHF</td>
<td>Yes</td>
<td>Eq. (18)</td>
<td>Eq. (19)</td>
</tr>
<tr>
<td>Case 2</td>
<td>SHF</td>
<td>No</td>
<td>Eq. (18)</td>
<td>Eq. (19)</td>
</tr>
<tr>
<td>Case 3</td>
<td>GHF</td>
<td>No</td>
<td>Eq. (18)</td>
<td>Eq. (19)</td>
</tr>
<tr>
<td>Case 4</td>
<td>SHF</td>
<td>No</td>
<td>Eq. (20)</td>
<td>-</td>
</tr>
<tr>
<td>Liovic et al. (2010)</td>
<td>SHF</td>
<td>-</td>
<td>Eq. (20)</td>
<td>-</td>
</tr>
</tbody>
</table>
Table 2: Spurious velocity and curvature standard deviation for a stationary droplet at $t/T_v = 1$. *Up to round-off errors.

<table>
<thead>
<tr>
<th>$R/\Delta$</th>
<th>Spurious velocity $(L_2)$</th>
<th>Spurious velocity $(L_\infty)$</th>
<th>Curvature standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$5.754 \times 10^{-2}$</td>
<td>$2.030 \times 10^{-1}$</td>
<td>$2.417 \times 10^{-4}$</td>
</tr>
<tr>
<td>4</td>
<td>$1.582 \times 10^{-2}$</td>
<td>$7.522 \times 10^{-2}$</td>
<td>$1.031 \times 10^{-4}$</td>
</tr>
<tr>
<td>8</td>
<td>$6.464 \times 10^{-3}$</td>
<td>$1.277 \times 10^{-3}$</td>
<td>$1.670 \times 10^{-4}$</td>
</tr>
<tr>
<td>12</td>
<td>$7.600 \times 10^{-3}$</td>
<td>$2.842 \times 10^{-3}$</td>
<td>$1.319 \times 10^{-3}$</td>
</tr>
<tr>
<td>16</td>
<td>$0^*$</td>
<td>$0^*$</td>
<td>$0^*$</td>
</tr>
</tbody>
</table>
Table 3: Error in time period and interface location with different grid resolutions for an oscillating droplet.

<table>
<thead>
<tr>
<th>$R_0/\Delta$</th>
<th>Time period error ($%$)</th>
<th>Interface location error ($L_2$)</th>
<th>Interface location error ($L_\infty$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>$6.956 \times 10^{-1}$</td>
<td>$8.101 \times 10^{-2}$</td>
<td>$1.633 \times 10^{-1}$</td>
</tr>
<tr>
<td>16</td>
<td>$4.283 \times 10^{-1}$</td>
<td>$2.899 \times 10^{-2}$</td>
<td>$5.361 \times 10^{-2}$</td>
</tr>
<tr>
<td>32</td>
<td>$1.309 \times 10^{-1}$</td>
<td>$1.062 \times 10^{-2}$</td>
<td>$3.141 \times 10^{-2}$</td>
</tr>
</tbody>
</table>
Table 4: Details of the simulations for a rising bubble under gravity.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computational grid $(n_x, n_y, n_z)$</td>
<td>(320, 100, 100)</td>
<td>cell</td>
</tr>
<tr>
<td>Droplet initial diameter $(D)$</td>
<td>20</td>
<td>cell</td>
</tr>
<tr>
<td>Droplet initial position $(x_0, y_0, z_0)$</td>
<td>(20, 50, 50)</td>
<td>cell</td>
</tr>
<tr>
<td>$E_0$</td>
<td>0.1 to 40</td>
<td>-</td>
</tr>
<tr>
<td>$\log(Mo)$</td>
<td>-11 to 1</td>
<td>-</td>
</tr>
<tr>
<td>Viscosity ratio $(\mu_2/\mu_1)$</td>
<td>100</td>
<td>-</td>
</tr>
<tr>
<td>Density ratio $(\rho_2/\rho_1)$</td>
<td>100</td>
<td>-</td>
</tr>
</tbody>
</table>
Table 5: Bubble terminal rise velocity for a wide range of \( Eo \) and \( Mo \). The present hybrid CV-GHF method is compared with experimental results of [Grace 1973] (G) and numerical results of [Baltussen et al. 2014] from CV, HF and TF methods. − is used when the particular curvature finding method fails.

<table>
<thead>
<tr>
<th>Case</th>
<th>( Eo )</th>
<th>( \log(Mo) )</th>
<th>G</th>
<th>CV</th>
<th>HF</th>
<th>TF</th>
<th>CV-GHF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-7</td>
<td>( 8.0 \times 10^{-7} )</td>
<td>( 8.1 \times 10^{-7} )</td>
<td>( 7.9 \times 10^{-7} )</td>
<td>( 8.0 \times 10^{-7} )</td>
<td>( 8.157 \times 10^{-7} )</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>-3</td>
<td>( 1.8 \times 10^{-3} )</td>
<td>( 2.0 \times 10^{-3} )</td>
<td>( 2.0 \times 10^{-3} )</td>
<td>( 2.0 \times 10^{-3} )</td>
<td>( 1.996 \times 10^{-3} )</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>-1</td>
<td>( 1.8 \times 10^{-1} )</td>
<td>( 2.2 \times 10^{-1} )</td>
<td>( 2.1 \times 10^{-1} )</td>
<td>( 1.9 \times 10^{-1} )</td>
<td>( 2.086 \times 10^{-1} )</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>-11</td>
<td>( 2.8 \times 10^{-4} )</td>
<td>( 2.2 \times 10^{-4} )</td>
<td>-</td>
<td>( 2.3 \times 10^{-4} )</td>
<td>( 2.774 \times 10^{-4} )</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>-9</td>
<td>( 9.0 \times 10^{-2} )</td>
<td>-</td>
<td>( 7.3 \times 10^{-2} )</td>
<td>( 7.4 \times 10^{-2} )</td>
<td>( 8.671 \times 10^{-2} )</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>-7</td>
<td>( 2.9 \times 10^{-2} )</td>
<td>-</td>
<td>( 2.3 \times 10^{-2} )</td>
<td>( 2.3 \times 10^{-2} )</td>
<td>( 2.757 \times 10^{-2} )</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>-3</td>
<td>( 2.4 \times 10^{-1} )</td>
<td>( 2.3 \times 10^{-1} )</td>
<td>( 2.3 \times 10^{-1} )</td>
<td>( 2.2 \times 10^{-1} )</td>
<td>( 2.353 \times 10^{-1} )</td>
</tr>
<tr>
<td>8</td>
<td>0.1</td>
<td>-11</td>
<td>( 1.4 \times 10^{-2} )</td>
<td>( 2.6 \times 10^{-2} )</td>
<td>( 2.5 \times 10^{-2} )</td>
<td>-</td>
<td>( 2.641 \times 10^{-2} )</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>-11</td>
<td>( 1.7 \times 10^{-3} )</td>
<td>( 1.6 \times 10^{-3} )</td>
<td>-</td>
<td>( 1.7 \times 10^{-3} )</td>
<td>( 1.851 \times 10^{-3} )</td>
</tr>
<tr>
<td>10</td>
<td>40</td>
<td>-11</td>
<td>( 7.0 \times 10^{-4} )</td>
<td>( 5.8 \times 10^{-4} )</td>
<td>( 5.9 \times 10^{-4} )</td>
<td>( 5.8 \times 10^{-4} )</td>
<td>( 6.933 \times 10^{-4} )</td>
</tr>
<tr>
<td>11</td>
<td>0.1</td>
<td>-7</td>
<td>( 4.6 \times 10^{-9} )</td>
<td>( 5.8 \times 10^{-9} )</td>
<td>( 5.5 \times 10^{-9} )</td>
<td>-</td>
<td>( 6.024 \times 10^{-9} )</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>-7</td>
<td>( 2.1 \times 10^{-2} )</td>
<td>-</td>
<td>( 1.9 \times 10^{-2} )</td>
<td>( 1.9 \times 10^{-2} )</td>
<td>( 1.891 \times 10^{-2} )</td>
</tr>
<tr>
<td>13</td>
<td>40</td>
<td>-7</td>
<td>( 7.2 \times 10^{-2} )</td>
<td>( 5.9 \times 10^{-2} )</td>
<td>( 5.7 \times 10^{-2} )</td>
<td>( 5.7 \times 10^{-2} )</td>
<td>( 6.728 \times 10^{-2} )</td>
</tr>
<tr>
<td>14</td>
<td>0.2</td>
<td>-3</td>
<td>( 1.6 \times 10^{-1} )</td>
<td>( 2.0 \times 10^{-1} )</td>
<td>( 1.9 \times 10^{-1} )</td>
<td>-</td>
<td>( 1.861 \times 10^{-1} )</td>
</tr>
<tr>
<td>15</td>
<td>40</td>
<td>-3</td>
<td>( 6.5 \times 10^{-1} )</td>
<td>( 6.0 \times 10^{-1} )</td>
<td>( 5.9 \times 10^{-1} )</td>
<td>( 6.0 \times 10^{-1} )</td>
<td>( 5.872 \times 10^{-1} )</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>1</td>
<td>( 1.4 \times 10^{-1} )</td>
<td>( 1.7 \times 10^{-1} )</td>
<td>( 1.7 \times 10^{-1} )</td>
<td>( 1.7 \times 10^{-1} )</td>
<td>( 1.668 \times 10^{-1} )</td>
</tr>
<tr>
<td>17</td>
<td>40</td>
<td>1</td>
<td>( 3.0 \times 10^{-9} )</td>
<td>( 3.7 \times 10^{-9} )</td>
<td>( 3.7 \times 10^{-9} )</td>
<td>( 3.8 \times 10^{-9} )</td>
<td>( 3.760 \times 10^{-9} )</td>
</tr>
</tbody>
</table>
Figure 1: Discretization of surface tension force in balanced-force concept. Volume fraction $F$ is stored at cell center (●). Surface tension force in $x$ ($F_{\sigma x}$) and $y$ ($F_{\sigma y}$) directions are discretized at face centers in $x$ (■) and $y$ (▲) directions, respectively.
Figure 2: Summation stencil to compute heights: A fixed 7 × 3 stencil for SHF (−) and an adaptive stencil for GHF (−−) methods. Different scenarios for the interfacial cell \((i, j)\) in SHF method: [a] height belongs to the cell [b] height doesn’t belong to the cell.
Figure 3: Comparison of SHF and GHF methods for nearby interfaces. A fixed $7 \times 3$ stencil for SHF (−) and an adaptive stencil for GHF (−−) are constructed around cell (□) for curvature computation.
Figure 4: Comparison of SHF and GHF methods based on the direction of HF stencil. A fixed 7 × 3 stencil for SHF (---) and an adaptive stencil for GHF (----) are constructed around cell (□) for curvature computation.
Figure 5: Convergence of $L_2$ and $L_\infty$ curvature error norms for case 1 (□), case 2 (△), case 3 (○), case 4 (□) and Liovic et al. 2010 (+). Details of each case is presented in Table 1. Second-order line (---) is shown as a reference.
Figure 6: Contour of the local curvature error ($\delta\kappa_i$) for (a) case 1, (b) case 2 and (c) case 3 for the spherical interface with $R/\Delta = 24$. Details of each case is presented in Table I.
Figure 7: Convergence of curvature error norms for the hybrid CV-GHF method: i) $L_2$ (△) and ii) $L_{\infty}$ governed by CV (○) and GHF (□). (−) shows the fraction of interfacial cells ($0 < F < 1$) using CV method for curvature computation. First-order (⋯) and second-order (−−) lines are shown as a reference.
Figure 8: Contour of the local curvature error ($\delta \kappa_i$) for the present CV method for the spherical interface with $R/\Delta = 24$. 
Figure 9: Results for a stationary droplet test: a) maximum spurious velocity ($L_{\infty}$) with dimensionless time ($t/T_v$) for $R/\Delta = 2 \, (+), \, 4 \, (--) , \, 8 \, (-\cdots -) , \, 12 \, (--) \, \text{and} \, 16 \, (--) $; b) curvature error norms: $L_{\infty}$ (○) and $L_2$ (□); c) pressure error $E(\Delta p)$ (○) and d) droplet shape error norms: $L_{\infty}$ (○) and $L_2$ (□). Note that b), c) and d) are plotted at $t/T_v = 1$ and second-order (--) line is shown as a reference.
Figure 10: Dimensionless interface location \( (r(t)/\eta) \) with dimensionless time \( (t/\tau) \) for an oscillating droplet. Analytical solution \( \text{(Eq. \( \cdot \cdot \cdot \)) by Lamb (1932)} \) and present numerical results with \( R_0/\Delta = 8 (\cdots), 16 (\cdots) \) and \( 32 (\cdots) \).
Figure 11: Local mesh refinement technique to initiate $F$ field. A single Cartesian computational cell $(i,j)$ is presented with $l = 2$ (two levels of refinement): level 1 (−) and level 2 (−). Circular interface (−) is approximated by a linear segment (−−) in the subdivisions. ● represents the cell center where $F$ needs to be initiated, ■ represents vertices of the cell/subdivision and × represents the intersection points of the interface with subdivision boundaries.
Figure 12: Convergence of the $L_2$ volume error norm (○) with grid resolution. Second-order (—) line is shown as a reference.