Commun. Comput. Phys.
doi: 10.4208/cicp.170510.290311a

Vol. **11**, No. 1, pp. 215-248
January 2012

A Preconditioned Implicit Free-Surface Capture Scheme for Large Density Ratio on Tetrahedral Grids

Xin Lv[1,*], Qingping Zou[2], D. E. Reeve[3] and Yong Zhao[4]

[1] *International Center for Numerical Methods in Engineering (CIMNE)-Singapore, 51 Science Park Road, #04-20, The Aries Singapore Science Park II, 117586, Singapore.*
[2] *Department of Civil and Environmental Engineering, University of Maine, Orono, Maine 04469, USA.*
[3] *Center for Coastal Dynamics and Engineering, School of Engineering, University of Plymouth, Devon PL4 8AA, United Kingdom.*
[4] *College of Engineering, Alfaisal University, Al Maathar Road, P.O. Box 50927, Riyadh 11533, Kingdom of Saudi Arabia.*

Received 17 May 2010; Accepted (in revised version) 29 March 2011

Communicated by Boo Cheong Khoo

Available online 8 September 2011

Abstract. We present a three dimensional preconditioned implicit free-surface capture scheme on tetrahedral grids. The current scheme improves our recently reported method [10] in several aspects. Specifically, we modified the original eigensystem by applying a preconditioning matrix so that the new eigensystem is virtually independent of density ratio, which is typically large for practical two-phase problems. Further, we replaced the explicit multi-stage Runge-Kutta method by a fully implicit Euler integration scheme for the Navier-Stokes (NS) solver and the Volume of Fluids (VOF) equation is now solved with a second order Crank-Nicolson implicit scheme to reduce the numerical diffusion effect. The preconditioned restarted Generalized Minimal RESidual method (GMRES) is then employed to solve the resulting linear system. The validation studies show that with these modifications, the method has improved stability and accuracy when dealing with large density ratio two-phase problems.

AMS subject classifications: 65E05

Key words: VOF, level set, free surface, unstructured finite volume method, implicit method, restarted GMRES, tetrahedral grid.

*Corresponding author. *Email addresses:* lvxin@cimne-sgp.com (X. Lv), qingping.zou@maine.edu (Q.-P. Zou), dominic.reeve@plymouth.ac.uk (D. E. Reeve), zyong@alfaisal.edu (Y. Zhao)

1 Introduction

We have recently reported a novel Coupled Level Set/VOF method for interfacial flow simulations on three dimensional unstructured tetrahedral grids [10]. At each time step, we evolve both the level set function and the volume fraction. The level set function is evolved by solving the level set advection equation using a high resolution characteristic based finite volume method. The volume fraction advection is performed using a bounded compressive Normalised Variable diagram (NVD) scheme. The interface is reconstructed using both the level set and the volume fraction information. In particular, the interface normal vector is calculated from the level set function while the intercepts are determined by enforcing mass conservation based on the volume fraction. The novelty of the method is that we use an analytic method to find the intercepts on tetrahedral grids, which makes interface reconstruction efficient and conserves volume of fluid exactly. The level set function is then reinitialized to the signed distance to the reconstructed interface. Furthermore, the adaptive combination of high resolution discretization schemes ensures the preservation of the sharpness and shape of the interface while retaining boundedness of the volume fraction field. Since the level set function is continuous, the interface normal vector calculation is straightforward and accurate compared to a classic volume-of-fluid method, while tracking the volume fraction is essential for enforcing mass conservation. The method is also coupled to a well validated finite volume based Navier-Stokes incompressible flow solver (Tetrinke) [11,12]. The code validation presented in [10] shows that the proposed method can conserve the mass very accurately and is able to maintain the sharpness of the interface. The coupling of level set and VOF is not the focus of this paper, the details of which can be found in [10]. In our earlier work [10], the discretized equations are marched forward in time using explicit schemes to reduce overall memory consumption. But this low memory demand comes with the sacrifices of stability and convergence speed. Therefore several convergence accelerating techniques including multigrid and implicit residual smoothing are needed to compensate the performance drop. Our numerical analysis has confirmed its good performance using several classical benchmark problems. However, the approximations and point implicit treatment introduced in that method to achieve low-cost computation can destroy the balance between the left-hand side and right-hand side of the equation and can thus slow the convergence rate when dealing with a stiff system (two-phase flow simulation with large density ratio is a good example). Moreover, such explicit schemes will be subject to CFL restrictions. In some simulations, time step size restriction can become a constraining factor, severely limiting the period of the flow simulation. Wave overtopping simulation in coastal engineering is a typical example of this kind, which requires thousands of waves to be simulated before reasonable statistical results can be readily obtained.

Another key feature of our previously proposed free-surface capture scheme is so called air-phase deactivation technique, which deactivates the gaseous phase computation if the density ratio between the two phases exceeds a specified magnitude. It is well

known that solving a two phase flow will pose difficulties when the density ratio between the liquid and gaseous phase is large. This obstacle mainly comes from the fact that in such a case, the pressure gradient distribution is discontinuous across the interface. This implies that any small pressure gradient that is transmitted from the liquid to the air region due to numerical error will accelerate the air considerably. This in turn will lead to inaccuracy, causing more spurious pressures. The whole cycle may, in fact, lead to a complete divergence of the solution. In the implementation presented in [10], a switch subroutine was designed and its function was monitoring the density ratio between the two phases. If the ratio is too large, say above 200, it automatically deactivates the computation for the air phase. All of the necessary information needed in the computation of the liquid phase is then extrapolated from inside the liquid phase itself. The position of deactivation interface serves as a tuneable parameter and enables the user to find a balance between the stability and accuracy. Generally, this parameter is between 0 and 0.5. A smaller number will ensure a more accurate result but the time step size must be reduced accordingly to make the solver stable. In our experience, this parameter can never take a value under 0.4 in the explicit framework. Technically, therefore, the method reported in [10] is a single-phase free surface capture scheme. The deactivation of the gaseous phase, which ensures robustness and decreases the total simulation time significantly, renders the method inappropriate to certain cases where the impact of the gaseous phase cannot be simply neglected or the interaction of the two phases is crucial. Furthermore, the deactivation of the gaseous phase in any level will cause physical inconsistency and thus affects the accuracy.

To date, numerous two-phase free surface capture schemes have been proposed in the literature. These methods have the merit handling very large density ratios without losing accuracy and stability. Among others, the Ghost Fluid Method (GFM) which is based on the level set method in its original form [28] has received much attention recently, and has been widely used by many researchers in practical applications of free surface flow computations. Essentially, GFM exploits the concept of ghost and real fluid cells and manages them with an overlapping Schwarzlike numerical procedure. In the material interface region, it sets the values of the pressure and normal velocity in the ghost fluid cells to those in the real fluid cells. To eliminate an otherwise spurious "over-heating" phenomenon, it computes the density of the ghost fluid using an isobaric fix technique. As explained in [28], this isobaric fix requires the solution of yet another auxiliary partial differential equation and therefore increases further the computational complexity of the method. Also as pointed out by some researchers (see [29] for example), the original GFM fails to solve some air/water problems of interest due to the large density ratio. In [31], Kang and the co-authors extended GFM to multiphase incompressible flow including the effects of viscosity, surface tension and gravity. The novelty of their approach is that they incorporated the boundary condition capturing approach for the variable coefficient Poisson equation developed in [30] to treat a sharp interface by means other than numerical smearing. They have showed improved accuracy and robustness of their approach against the original GFM.

In this study, we propose an alternative approach to overcome the above mentioned drawbacks by introducing a preconditioning technique and constructing a fully implicit scheme to alleviate the time step size restriction and improve robustness. In the remainder of this paper, we will describe the governing equations for incompressible fluid flow and the free surface evolution, the basic discretization technique used for tetrahedral meshes, and the linear system iteration solution procedure. The performance of the resulting algorithm and its comparisons against the GFM in three dimensions will be demonstrated by several selected numerical examples. A summary of major findings and conclusion is also given.

2 Governing equations and numerical methods

Here we consider incompressible flows with two different fluids. The density of one fluid is ρ_a and the density of the second fluid is ρ_g. The nondimensional governing 3D equations, modified by the artificial compressibility method (ACM) (see [16]), are given in non-dimensional vector form as,

$$\Gamma\frac{\partial \mathbf{W}}{\partial \tau}+\mathbf{K}\frac{\partial \mathbf{W}}{\partial t}+\nabla\cdot\vec{\mathbf{F}}_c=\nabla\cdot\vec{\mathbf{F}}_v+\vec{\mathbf{S}}, \tag{2.1}$$

where

$$\mathbf{W}=\begin{bmatrix} p\\ u\\ v\\ w\\ \varphi \end{bmatrix},\qquad \vec{\mathbf{F}}_c=\begin{bmatrix} \vec{U}\\ u\vec{U}+\dfrac{p}{\rho\delta_{ij}}\\ v\vec{U}+\dfrac{p}{\rho\delta_{ij}}\\ w\vec{U}+\dfrac{p}{\rho\delta_{ij}}\\ \varphi\vec{\mathbf{U}} \end{bmatrix},\qquad \vec{\mathbf{F}}_v=\begin{bmatrix} 0\\ \dfrac{\mu}{\rho Re}\nabla u\\ \dfrac{\mu}{\rho Re}\nabla v\\ \dfrac{\mu}{\rho Re}\nabla w\\ 0 \end{bmatrix},$$

$$\mathbf{K}=\begin{bmatrix} 0&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\\ 0&0&0&0&1 \end{bmatrix},\qquad \Gamma=\begin{bmatrix} \dfrac{1}{\beta\rho}&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\\ \dfrac{\varphi}{\beta\rho}&0&0&0&1 \end{bmatrix},\qquad \vec{\mathbf{S}}=\begin{bmatrix} 0\\ \dfrac{1}{\rho}F_{Sx}+F_{gx}\\ \dfrac{1}{\rho}F_{Sy}+F_{gy}\\ \dfrac{1}{\rho}F_{Sz}+F_{gz}\\ 0 \end{bmatrix}.$$

In all the equations above, $\mathbf{W}$ is the vector of dependent variables, the velocity vector is defined as $\vec{\mathbf{U}}=u\cdot\vec{i}+v\cdot\vec{j}+w\cdot\vec{k}$. p and ρ are pressure and density, respectively, β the constant parameter introduced by ACM (which is set to be 10 for all of the testing cases reported in this paper). $\vec{\mathbf{F}}_c$ and $\vec{\mathbf{F}}_v$ are the convective flux and viscous flux vectors. $\vec{\mathbf{S}}$ contains the surface tension and gravity terms. The first term on the left-hand side of Eq. (2.1) is a

partial derivative of pressure with respect to pseudo-time τ (the artificial compressibility term), which is introduced to couple velocity and pressure fields for the calculation of pressure based on the divergence-free condition. Γ is a preconditioning matrix that arises with the implementation of the artificial compressibility method. **K** is the unit matrix with its first element being zero, and t is the physical time. Since the surface tension and gravity play significant roles during the development of a free surface flow process, the combined effects for surface tension and gravity are included. F_g is the gravity force per unit mass and is given as

$$\vec{F}_g = \frac{\vec{n}_g}{Fr^2}, \tag{2.2}$$

where Fr is the Froude number and $\vec{n}_g$ the unit vector along the prescribed direction of gravity. $\vec{F}_s$ is the surface tension force per unit volume, given in [22]

$$\vec{F}_s = \frac{\kappa \nabla \varphi}{We}, \tag{2.3}$$

where κ is the curvature of the interface and We is the Weber number. The curvature can be readily computed from the continuous level set function field as

$$\kappa = \left(-\nabla \cdot \frac{\nabla \varphi}{|\nabla \varphi|} \right). \tag{2.4}$$

The level set function φ is defined to be a signed distance function

$$|\varphi(\vec{x})| = d(\vec{x}) = \min_{x_I \in I}(|\vec{x} - \vec{x}_I|), \tag{2.5}$$

where I is the VOF interface, $\varphi > 0$ on one side of the interface and $\varphi < 0$ on the other. In standard level set methods, the advection of φ, including a reinitialization step to retain φ as a signed distance function, is not done in a conservative way, not even for divergence free velocity fields. This implies that the total mass bounded by the zero level set is not conserved. This drawback has been addressed in our proposed method.

To represent density and viscosity discontinuities over the interface the Heaviside function:

$$H(\varphi) = 0, \qquad \varphi < 0, \tag{2.6a}$$
$$H(\varphi) = 1, \qquad \varphi > 0, \tag{2.6b}$$
$$H(\varphi) = 0.5, \quad \varphi = 0 \tag{2.6c}$$

is needed. In practical computations, to achieve numerical robustness, a smeared out version of Heaviside function is used,

$$H_\varepsilon(\varphi) = \begin{cases} 0, & \varphi < -\varepsilon, \\ \frac{1}{2} + \frac{\varphi}{2\varepsilon} + \frac{\varphi}{2\pi} \sin\left(\frac{\pi \varphi}{\varepsilon}\right), & -\varepsilon \le \varphi \le \varepsilon, \\ 1, & \varphi > \varepsilon, \end{cases} \tag{2.7}$$

where ε corresponds to half the thickness of the interface. It should not be too small for the reason of robustness. In our proposed method, $1\sim2$ times of the smallest grid size has been found to be sufficient to keep the code stable and accurate.

Finally, the constitutive relations for the density and dynamic viscosity are defined to close all of the above equations

$$\rho=H_\varepsilon(\varphi)\rho_a+(1-H_\varepsilon(\varphi))\rho_g,\qquad \mu=H_\varepsilon(\varphi)\mu_a+(1-H_\varepsilon(\varphi))\mu_g. \tag{2.8}$$

The total density conservation equation is replaced by the level set equation, and they are equivalent if ρ_a and ρ_g are constant.

The eigenvalue system of the preconditioned equations is provided in Appendix A. In the development of the baseline differential system presented above, a number of physical, numerical and practical issues were considered [1]. Firstly, a corresponding artificial time-derivative term $(\frac{\varphi}{\beta\rho}\frac{\partial p}{\partial\tau})$ is introduced in the level set equation, which ensures that the proper differential equation (in non-conservative form) is satisfied. To illustrate this, consider the level set equation, which includes these artificial time-derivative terms:

$$\frac{\varphi}{\beta\rho}\frac{\partial p}{\partial\tau}+\frac{\partial\varphi}{\partial\tau}+\frac{\partial\varphi}{\partial t}+\nabla\cdot(\varphi\vec{\mathbf{U}})=0. \tag{2.9}$$

Expanding the convection term in Eq. (2.9) and substituting the continuity equation in a form which isolates the divergence of the velocity field

$$\nabla\cdot\vec{\mathbf{U}}=-\frac{1}{\beta\rho}\frac{\partial p}{\partial\tau} \tag{2.10}$$

yields after some simplification,

$$\frac{\partial\varphi}{\partial\tau}+\frac{\partial\varphi}{\partial t}+\vec{\mathbf{U}}\cdot\nabla\varphi=0, \tag{2.11}$$

which is the proper non-conservative form for the level set convection equation.

Secondly, the requirement that the eigensystem is independent of both the density ratio and the level set function will lead to a performance that is commensurate with that of single-phase problems for a wide range of multi-phase conditions. These considerations give rise to the preconditioned system in Eq. (2.1), where $(\beta\rho)^{-1}$ is chosen as the preconditioning parameter rather than more commonly used β^{-1}, and $\frac{\varphi}{\beta\rho}\frac{\partial p}{\partial\tau}$ is added to the level set equation. Accordingly, the eigenvalues of the inviscid equations are independent of the level set function and density ratio. This is not the case for other choices of preconditioning matrix, Γ. For example, not including the terms $\frac{\varphi}{\beta\rho}\frac{\partial p}{\partial\tau}$ in the level set equation or selecting β^{-1} as the preconditioning parameter yields the eigenvalues,

$$\tilde{\Lambda}=\left[U+c,U-c,U,U,U\right],\qquad c=\left(U^2+\frac{\beta}{\rho}\right)^{\frac{1}{2}}, \tag{2.12}$$

where the nondimensional density ρ arises in the eigenvalues which means the equation system is still density ratio dependent. The local time-steps and matrix dissipation operators presented below are derived from the inviscid multi-phase eigensystem, which has been shown to be closely related to the known single-phase eigensystem. This has had the practical advantage of making the single-phase predecessor code easier to adapt to the multi-phase system.

Following [10], Eq. (2.1) can be recast in an integral form as follows,

$$\Gamma\frac{\partial}{\partial\tau}\iiint_v \mathbf{W}dV+\mathbf{K}\frac{\partial}{\partial t}\iiint_v \mathbf{W}dV+\iiint_v\left[\nabla\cdot(\vec{\mathbf{F}}_c-\vec{\mathbf{F}}_v)+\vec{\mathbf{S}}\right]dV=0. \tag{2.13}$$

Eq. (2.13) is equivalent to the following equation,

$$\Gamma\frac{\partial}{\partial\tau}\iiint_v \mathbf{W}dV+\mathbf{K}\frac{\partial}{\partial t}\iiint_v \mathbf{W}dV+\oint_{S_{cv}}(\vec{\mathbf{F}}_c-\vec{\mathbf{F}}_v)\cdot dS+\iiint_v\vec{\mathbf{S}}dV=0. \tag{2.14}$$

Once the artificial steady state is reached, the derivatives with respect to τ become zero and the above equation reduces to the following equation,

$$\mathbf{K}\frac{\partial}{\partial t}\iiint_v \mathbf{W}dV+\oint_{S_{cv}}(\vec{\mathbf{F}}_c-\vec{\mathbf{F}}_v)\cdot dS+\iiint_v\vec{\mathbf{S}}dV=0. \tag{2.15}$$

Eq. (2.15) shows that the preconditioning matrix does not affect the solution and the original unsteady incompressible Navier-Stokes equations are fully recovered. In this work, a cell-vertex edge based finite volume scheme is adopted. For every vertex q, a control volume cv is constructed using the median duals of the tetrahedral cells. Following the finite volume space-integration procedure introduced in [10–12] and using the Euler implicit time-integration, Eq. (2.14) can be written in discrete form as

$$\Gamma_q\Delta V_{cv,q}^{n+1}\frac{\Delta\mathbf{W}}{\Delta\tau}=-R_q^{n+1,m+1}-\mathbf{K}_q\Big(\frac{1.5\Delta V_{cv,q}^{n+1}\mathbf{W}_q^{n+1,m+1}-2.0\Delta V_{cv,q}^{n}\mathbf{W}_q^{n}+0.5\Delta V_{cv,q}^{n-1}\mathbf{W}_q^{n-1}}{\Delta t}\Big), \tag{2.16}$$

where subscript q means the quantities are related to the vertex q (from rest of this paper, this subscript will be discarded for the sake of simplicity). $\Delta V_{cv,q}^{n+1}$ stands for the volume of the current control volume surrounding vertex q at $n+1$ physical time level. Δt and $\Delta\tau$ are the physical and pseudo time step size respectively and ΔW the difference of unknown vector at between pseudo time level m and $m+1$; i.e.,

$$\Delta\mathbf{W}=\mathbf{W}^{n+1,m+1}-\mathbf{W}^{n+1,m}. \tag{2.17}$$

The first term on the right hand side of Eq. (2.16), the total residual including convective and viscous fluxes, as well as source terms is defined as

$$R^{n+1,m+1}=\Big\{\sum_{j=1}^{nbseg}\left[(\vec{F}_c)_{qj}\cdot\vec{N}\Delta S\right]_q-\frac{1}{3}\sum_{j=1}^{ncell}\left(\vec{F}_v\cdot\vec{M}\Delta S_q\right)_j-S\Delta V_{q,cv}\Big\}^{n+1,m+1} \tag{2.18}$$

and is evaluated at $n+1$ physical and $m+1$ pseudo time step, where $\vec{N}=\{n_x,n_y,n_z\}$ and ΔS are the normal vector and area of the control volume boundary face associated with the center of edge qj respectively. Similarly, $\vec{M}=\{m_x,m_y,m_z\}$ and ΔS_q are the normal vector and area of the control volume boundary face associated with the center of cell j. *nbseg* and *ncell* are the total number of edges and tetrahedral cells associated with node q. Eq. (2.16) can be linearized in pseudo time as

$$\underbrace{\Big(K\Big(\frac{1.5\Delta V_{cv}^{n+1}}{\Delta t}\Big)+\Gamma\Big(\frac{\Delta V_{cv}^{n+1}}{\Delta\tau}\Big)+\hat{A}_c-\hat{A}_v-\hat{A}_s\Delta V_{cv}^{n+1}\Big)}_{\text{Left-hand side Jacobian}}(W^{n+1,m+1}-W^{n+1,m})$$
$$=-R^{n+1,m}-K\Big(\frac{1.5\Delta V_{cv}^{n+1}\mathbf{W}^{n+1,m}-2.0\Delta V_{cv}^{n}\mathbf{W}^{n}+0.5\Delta V_{cv}^{n-1}\mathbf{W}^{n-1}}{\Delta t}\Big), \tag{2.19}$$

where $K(1.5\Delta V_{cv}^{n+1}/\Delta t)$ emerges from the linearization of physical time term $K(1.5\Delta V_{cv}^{n+1}W^{n+1,m+1}/\Delta t)$ and $R^{n+1,m}$ is the total residual defined in Eq. (2.16) but evaluated at m pseudo time level. $\hat{A}_c$, $\hat{A}_v$ and $\hat{A}_s$ are the convective, viscous and source term Jacobian matrices at m pseudo time level due to the linearization procedure and the derivations of which will be described next. In this study, the source term Jacobian $\hat{A}_s$ is ignored.

In this work, Roe's flux difference splitting formula has been used to evaluate the numerical convection flux at the Control Volume's boundary face:

$$F_c=\frac{1}{2}\big[F_c(W^L)+F_c(W^R)-\Gamma|\Gamma^{-1}A_c(W^{Roe})|(W^R-W^L)\big]. \tag{2.20}$$

In Eq. (2.20), the superscripts R and L indicate the right and left states respectively. The flux differencing term in Eq. (2.20), $\Gamma|\Gamma^{-1}A_c(W^{Roe})|(W^R-W^L)$, can be recast in the form of $\Gamma|\Gamma^{-1}A_c(W^{Roe})|\Delta W$ and the full derivative of this term with respect to the solution vector (in general form) is

$$\frac{\partial\big[\Gamma|\Gamma^{-1}A_c(W^{Roe})|\Delta W\big]}{\partial W_i}=\frac{\partial\big[\Gamma|\Gamma^{-1}A_c(W^{Roe})|\big]}{\partial W_i}\Delta W+\Gamma|\Gamma^{-1}A_c(W^{Roe})|\frac{\partial[\Delta W]}{\partial W_i}, \tag{2.21}$$

where the index i represents either control volume i or its direct neighbour. Differentiating $\partial\big[\Gamma|\Gamma^{-1}A_c(W^{Roe})|\big]/\partial W_i$ produces third-rank tensors which are not only difficult to derive but are also quite expensive to compute. Barth [2] has found the full derivative of $\partial\big[\Gamma|\Gamma^{-1}A_c(W^{Roe})|\Delta W\big]/\partial W_i$ with some clever modifications to eliminate the tensor computations, reducing the complexity of the Jacobian computation to some degree. Through spectral radius analysis for 1-D flow he showed that for a smooth flow the approximate Jacobian is reasonably accurate up to $CFL=1000$ or even above. However, for the shock tube problem the difference between the true Jacobian and the approximate Jacobian grows after $CFL=10$, and becomes noticeable after $CFL=100$, showing that the approximate Jacobian will not be accurate enough for larger CFL numbers. This result is consistent with what we would expect from inspection of Eq. (2.21). For a smooth flow, ΔW-the difference in the two reconstructed solutions at a Gauss point-is on

the order of truncation error, so $(\partial\left[\Gamma|\Gamma^{-1}A_c(W^{Roe})|\right]/\partial W_i)\Delta W$ is very small compared to $\Gamma|\Gamma^{-1}A_c(W^{Roe})|\partial[\Delta W]/\partial W_i$, and the resulting approximate Jacobian will be acceptable. Near a discontinuity, however, this approximation is not accurate anymore, because $\partial\left[\Gamma|\Gamma^{-1}A_c(W^{Roe})|\right]/\partial W_i$ and ΔW will be $\mathcal{O}(1)$. Even though ignoring variations of $\Gamma|\Gamma^{-1}A_c(W^{Roe})|$ (treating it as a constant) introduces an error in the Jacobian, and thus causes stability problems when the CFL number is big. But in our situation a CFL number smaller than 5 is necessary in order to evolve the volume fraction equation correctly. The cause of this restriction will be explained shortly. This simplification of the Jacobian of Roe's flux leads to the linearized form of Eq. (2.20) as:

$$F_c^{m+1}=F_c^m+\underbrace{\frac{1}{2}\left(A_c(W^L)+\Gamma|\Gamma^{-1}A_c(W^{Roe})|\right)\Delta W^L+\frac{1}{2}\left(A_c(W^R)-\Gamma|\Gamma^{-1}A_c(W^{Roe})|\right)\Delta W^R}_{\hat{A}_c\Delta W}, \quad (2.22)$$

These two parts will be moved to the LHS of the discretized equation.

where

$$A_c=\frac{\partial F_c}{\partial W},$$

and has been defined in the appendix. As pointed out in [1], using the reconstructed values in each control volume at the Gauss point produces a more effective preconditioning matrix; in this paper, we also use linear reconstruction data in computing the Jacobian. Because the reconstruction has already been computed, the additional cost of using this more accurate data is negligible. Also, with this approach we include the effects of boundary conditions in the approximate Jacobian by computing the Jacobian of the boundary flux with respect to the solution in the interior of the domain. In this case, only the A_c term in the Jacobian need be computed. In our experience, the CPU time demanded for one approximate analytic Jacobian evaluation is $0.4-0.5$ of that of a third-order residual evaluation; reconstruction and limiting costs are not included here, as the limited reconstruction from the residual evaluation is simply re-used. In Eq. (2.22),

$$|\Gamma^{-1}A_c(W^{Roe})|=\hat{T}|\hat{\Lambda}|\hat{T}^{-1},$$

where T is the matrix whose columns are the right eigenvectors of $\hat{A}_c=\Gamma^{-1}A_c(W^{Roe})$ and $\hat{\Lambda}$ is a diagonal matrix whose elements are the absolute values of the eigenvalues of $\hat{A}_c$ (Check Appendix for their detailed definition). The matrix $\hat{A}_c$, dependent on left and right states at the interface, is evaluated using Roe-average variables. In the two-phase fluid system under consideration, the appropriate Roe average variables are given by [13] as:

$$\lambda=\frac{\sqrt{\rho^R}}{\sqrt{\rho^L}},\qquad \rho^{Roe}=\lambda\rho^L,\qquad p^{Roe}=\frac{p^L+p^R}{2}, \quad (2.23a)$$

$$\varphi^{Roe}=\frac{\varphi^L+\varphi^R}{2},\qquad \left(u^{Roe},v^{Roe},w^{Roe}\right)=\frac{(u,v,w)^R\lambda+(u,v,w)^L}{1+\lambda}, \quad (2.23b)$$

where density ρ is computed from Eq. (2.8). The left and right state vectors W^L and W^R at a control volume surface are evaluated using a nominally third-order upwind-biased interpolation scheme [10–12]. The viscous flux vector can be expressed in discretized form as:

$$F_v=\frac{\mu}{\rho Re}\begin{bmatrix} 0 \\ 2\frac{\partial u}{\partial x}n_x+\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)n_y+\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right)n_z \\ 2\frac{\partial v}{\partial y}n_y+\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)n_x+\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)n_z \\ 2\frac{\partial w}{\partial z}n_z+\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right)n_x+\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)n_y \\ 0 \end{bmatrix}$$

$$=\frac{\mu}{9\Delta V_{cv}\rho Re}\begin{bmatrix} 0 \\ 2\Big(\sum_{i=1}^{4}u_i(d_{sx})_i\Big)n_x+\Big(\sum_{i=1}^{4}u_i(d_{sy})_i+\sum_{i=1}^{4}v_i(d_{sx})_i\Big)n_y+\Big(\sum_{i=1}^{4}u_i(d_{sz})_i+\sum_{i=1}^{4}w_i(d_{sx})_i\Big)n_z \\ 2\Big(\sum_{i=1}^{4}v_i(d_{sy})_i\Big)n_y+\Big(\sum_{i=1}^{4}u_i(d_{sy})_i+\sum_{i=1}^{4}v_i(d_{sx})_i\Big)n_x+\Big(\sum_{i=1}^{4}v_i(d_{sz})_i+\sum_{i=1}^{4}w_i(d_{sy})_i\Big)n_z \\ 2\Big(\sum_{i=1}^{4}w_i(d_{sz})_i\Big)n_z+\Big(\sum_{i=1}^{4}u_i(d_{sz})_i+\sum_{i=1}^{4}w_i(d_{sx})_i\Big)n_x+\Big(\sum_{i=1}^{4}v_i(d_{sz})_i+\sum_{i=1}^{4}w_i(d_{sy})_i\Big)n_y \\ 0 \end{bmatrix}. \quad (2.24)$$

Then the viscous flux at $m+1$ time level can be linearized as

$$F_v^{m+1}=F_v^m+\frac{\partial F_v^m}{\partial W^1}\Delta W^1+\frac{\partial F_v^m}{\partial W^2}\Delta W^2+\frac{\partial F_v^m}{\partial W^3}\Delta W^3+\frac{\partial F_v^m}{\partial W^4}\Delta W^4$$

$$=F_v^m+\frac{\mu}{9\Delta V_{cv}\rho Re}\sum_{i=1}^{4}\underbrace{\left\{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2(d_{sx})_in_x+(d_{sy})_in_y+(d_{sz})_in_z & (d_{sx})_in_y \\ 0 & (d_{sy})_in_x & 2(d_{sy})_in_y+(d_{sx})_in_x+(d_{sz})_in_z \\ 0 & (d_{sz})_in_x & (d_{sz})_in_y \\ 0 & 0 & 0 \end{bmatrix}\right.}$$

$$\underbrace{\left.\begin{bmatrix} 0 & 0 \\ (d_{sx})_in_z & 0 \\ (d_{sy})_in_z & 0 \\ 2(d_{sz})_in_z+(d_{sx})_in_x+(d_{sy})_in_y & 0 \\ 0 & 0 \end{bmatrix}\Delta W^i\right\}}_{\hat{A}_V\Delta W}. \quad (2.25)$$

These terms will be moved to the LHS of the discretized equation.

In Eqs. (2.24)-(2.25), $\vec{n}_S=\{n_x,\ n_y,\ n_z\}$ refers to the surface area vector in cell face normal direction and $(d_{\vec{S}})=\{d_{Sx},\ d_{Sy},\ d_{Sz}\}$ the surface vector that is opposite to node i of the tetrahedron being considered.

The evolution of the volume fraction (VOF) field is governed by

$$\frac{\partial F}{\partial t}+u\cdot\nabla F=0. \quad (2.26)$$

Due to the fact of $\nabla\cdot\vec{U}=0$ for incompressible flow, Eq. (2.26) can be reformulated as

$$\frac{\partial F}{\partial t}+\nabla(F\cdot\vec{U})=0. \tag{2.27}$$

This equation is in divergence form and can be easily discretized using control volume schemes. Using Euler implicit time-integration, and after performing implicit linearization, Eq. (2.27) can be written in discrete form as

$$\underbrace{\Big(\frac{\Delta V_{cv}^{n+1}}{\Delta\tau}+\frac{1.5\Delta V_{cv}^{n+1}}{\Delta t}+A_F\Big)}_{\text{Left-hand side Jacobian}}\Delta F=-R^{n+1,m}-\Big(\frac{1.5\Delta V_{cv}^{n+1}F^{n+1,m}-2.0\Delta V_{cv}^{n}F^{n}+0.5\Delta V_{cv}^{n-1}F^{n-1}}{\Delta t}\Big). \tag{2.28}$$

Following the derivations for Eq. (2.19), the total residual including only convective fluxes is defined as

$$R^{n+1,m+1}=\Big\{\sum_{j=1}^{nbseg}\big[(\vec{F}_c)_{qj}\cdot\vec{N}\Delta S\big]_q\Big\}^{n+1,m+1}$$

and is evaluated at $n+1$ physical and $m+1$ pseudo time step. The Jacobian due to implicit linearization A_F will be derived as follows.

As discussed in [10], the CICSAM (Compressive Interface Capturing Scheme for Arbitrary Meshes) developed by Ubbink and Issa [14] is employed for solving the VOF equation due to its good performance to maintain the sharpness of the interface while keeping reasonable accuracy. It makes use of the NVD concept (see [15]) and switches between different high resolution differencing schemes to yield a bounded scalar field, but one which preserves both the smoothness of the interface and its sharp definition (over one or two computational cells).

According to this method, the convection flux (or residual) at the Control Volume's boundary face is evaluated as

$$F_c^{m+1}=\big\{U\cdot F_{face}^{m+1}\big\}=\Big\{U\cdot\Big((1-\beta)\frac{F_{donor}^{m+1}+F_{donor}^{m}}{2}+\beta\frac{F_{acceptor}^{m+1}+F_{acceptor}^{m}}{2}\Big)\Big\}, \tag{2.29a}$$

$$U=un_x+vn_y+wn_z, \tag{2.29b}$$

where β is CICSAM weighting factor introduced in [14] and the contra-variant velocity U is assumed to be constant during current time step. Note that, as suggested by Ubbink and Issa [14], to minimize the numerical diffusion effect, a 2$^{\text{nd}}$ order Crank-Nicolson time-integration scheme is used here. Thus, the residual at $m+1$ pseudo time level can be linearised as

$$\begin{aligned}R^{n+1,m+1}&=R^{n+1,m}+\frac{\partial R}{\partial F_{donor}}\frac{\partial F_{donor}}{\partial\tau}\Delta\tau+\frac{\partial R}{\partial F_{acceptor}}\frac{\partial F_{acceptor}}{\partial\tau}\Delta\tau\\&\approx R^{n+1,m}+\frac{\partial R}{\partial F_{donor}}\Delta F_{donor}+\frac{\partial R}{\partial F_{acceptor}}\Delta F_{acceptor}.\end{aligned} \tag{2.30}$$

The Jacobians for donor and acceptor CVs can be evaluated as

$$A_{F,donor}=\frac{\partial R}{\partial F_{donor}}=\frac{1}{2}(1-\beta)U, \qquad A_{F,acceptor}=\frac{\partial R}{\partial F_{acceptor}}=\frac{1}{2}\beta U. \tag{2.31}$$

Recall that we mentioned earlier in our situation the CFL number may not exceed 5 in order to keep the evolution of the VOF function accurate. This restriction actually stems from the CICSAM method. The donor-acceptor concept [19, 20] forms the basis of this compressive differencing scheme, and it requires that a given fluid particle does not travel across one whole cell in one time step.

As suggested by some other researchers, for an implicit high order scheme, in order to achieve rapid convergence, it is necessary to make the boundary procedure implicit [9, 17]. Explicit treatments will deteriorate the convergence rate and sometimes lead to instability. In this study, a similar procedure to [17] is adopted. Consider the free-slip wall (which is also the most common boundary type one would encounter in free-surface simulations) as an example. At this boundary, only the pressure term will contribute to the convection flux computation:

$$F_c|_{Slip-Wall}=\begin{bmatrix}0\\ \rho^{-1}pn_x\\ \rho^{-1}pn_y\\ \rho^{-1}pn_z\\ 0\end{bmatrix} \tag{2.32}$$

and this flux can be linearized in pseudo time as

$$\begin{aligned}F^{m+1}_{c_{|_{Slip-Wall}}}&=F^{m}_{c_{|_{Slip-Wall}}}+\frac{\partial F^{m}_{c_{|_{Slip-Wall}}}}{\partial W}\nabla W\\ &=F^{m}_{c_{|_{Slip-Wall}}}+\underbrace{\begin{bmatrix}0&0&0&0&0\\ \rho^{-1}n_x&0&0&0&0\\ \rho^{-1}n_y&0&0&0&0\\ \rho^{-1}n_z&0&0&0&0\\ 0&0&0&0&0\end{bmatrix}\Delta W}_{\hat{A}_{bc}\Delta W}.\end{aligned} \tag{2.33}$$

This term will be moved to the LHS.

At the far field, the fluxes are computed using characteristic boundary condition, and they can be linearized similarly.

3 The linear system solver

As mentioned previously, the left and right states on CV faces are evaluated using 3rd order MUSCL schemes [11] in order to achieve higher-order fluxes computation. However, only a first-order representation of the numerical fluxes is considered while performing implicit linearization. This results in a greatly simplified sparse Jacobian matrix with a

graph identical to the graph of the supporting unstructured mesh. The penalty for making these approximations in the linearization process is that the quadratic convergence of Newton's method can no longer be achieved because of the mismatch and inconsistency between the right- and left-hand sides in the update equation. Although the number of time steps (Newton iterations, if CFL number tends to infinity) may increase, the cost per each time step is significantly reduced: it takes less CPU time to compute the Jacobian matrix and the conditioning of the simplified Jacobian matrix is improved, thus reducing computational cost to solve the resulting linear system. Another potential problem is that the viscous Jacobian evaluated with Eq. (2.25), when assembled into the LHS Jacobian, could make the system ill-conditioned, and thus the resultant linear system will become extremely hard to solve using a traditional iterative solver. But simply ignoring it leads to a degeneration of the convergence performance. One possible solution is ignoring the viscous Jacobian in the several initial steps of the linear solver, and then taking it into account in the following iterations.

Eq. (2.19) and (2.28) both represents a system of linear simultaneous algebraic equations and needs to be solved at each iteration step. They can be recast to the form of

$$Ax = b, \tag{3.1}$$

where A represents the left-hand side sparse Jacobian matrix with each entry being a $neqns*neqns$ square matrix, for Eq. (2.19), the left-hand-side Jacobian matrix is defined as

$$A = \Big(K\Big(\frac{1.5\Delta V_{cv}^{n+1}}{\Delta t}\Big) + \Gamma\Big(\frac{\Delta V_{cv}^{n+1}}{\Delta \tau}\Big) + \hat{A}_c - \hat{A}_v - \hat{A}_s \Delta V_{cv}^{n+1}\Big).$$

The number of nonzero entries in each row of the Jacobian matrix is related to the number of edges incident to the node associated with that row. In other words, each edge i, j will guarantee nonzero entries in the i-th column and j-th row and, similarly, the j-th column and i-th row. In addition, nonzero entries will be placed on the diagonal of the matrix representing each of the node point in the computational domain. Using an edge-based data structure, the left-hand side Jacobian matrix A is stored in sparse triplet matrix format, only non-zero entries are stored in a compressed manner. The symbol x represent the unknown vector W defined in Eq. [1] to be solved and b is the right-hand-side evaluated at m pseudo time level, for Eq. [19],

$$b = -R^{n+1,m} - K\Big(\frac{1.5\Delta V_{cv}^{n+1} W^{n+1,m} - 2.0\Delta V_{cv}^{n} W^{n} + 0.5\Delta V_{cv}^{n-1} W^{n-1}}{\Delta t}\Big).$$

The most widely used methods to solve Eq. (3.1) are iterative solution methods and approximate factorization methods. In this work, the generalized minimal residual (GMRES) method of Saad and Schultz [3] is used because of its excellent performance when dealing with a stiff system where the coefficient matrix is not symmetric and/or positive definite. The use of GMRES combined with different preconditioning techniques is becoming widespread in the CFD community for the solution of the Euler and Navier-Stokes equations [4–7]. Regarding this method, readers are referred to [3, 8, 9, 18, 27] for more details.

4 Numerical results

4.1 Breaking dam problem

As the first selected benchmark problem in this work, the collapse of a liquid column (see [19–21]) will firstly be examined. Measurements of the exact interface shape are not available, but some secondary data such as the speed of the wave front and the reduction of the column height are available [22]. Using this case, we will examine the efficiency and accuracy of currently proposed method. Several commonly used techniques for solving the linear system due to discretization will be compared in terms of CPU time (efficiency). The effect of the choice of the *position of deactivation interface* will be further discussed. The results for different density ratios will also be presented to demonstrate the capability of current solver when dealing with big density ratio problems. Furthermore, the comparison of current method and the Ghost Fluid Method (GFM, see [26]) in terms of total CPU time will also be given. Fig. 1 shows a schematic view of the domain setup which is used for the current flow prediction. The water column is initially supported on the right by a vertical plate drawn up rapidly at time $t = 0.0$s. Gravitational acceleration causes the water column in the left of the tank to seek the lowest possible level of potential energy. Thus, the column will collapse and eventually come to rest and occupy the bottom of the tank. The initial stages of the flow are dominated by inertia forces with viscous effects increasing rapidly as the water comes to rest. On such a large scale, the effect of surface tension forces is unimportant. For the numerical calculation no-slip boundary conditions have been applied to the bottom and sides of the tank. This experiment models a two-dimensional effect; therefore slip conditions have been applied to the front and back face of the tank. It should be noted that this is a two-phase simulation, i.e., both the water and gaseous phases are computed.

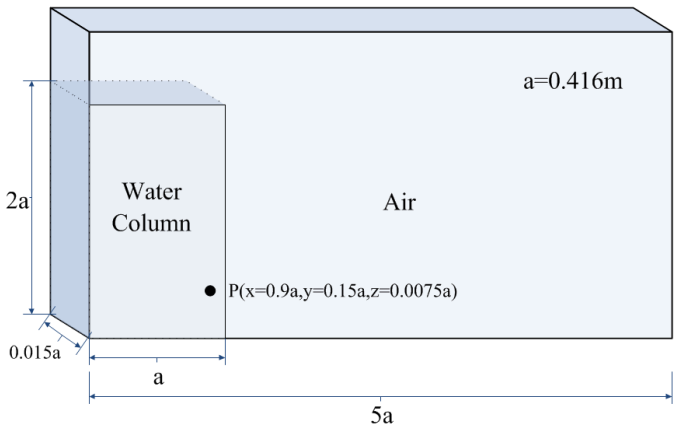
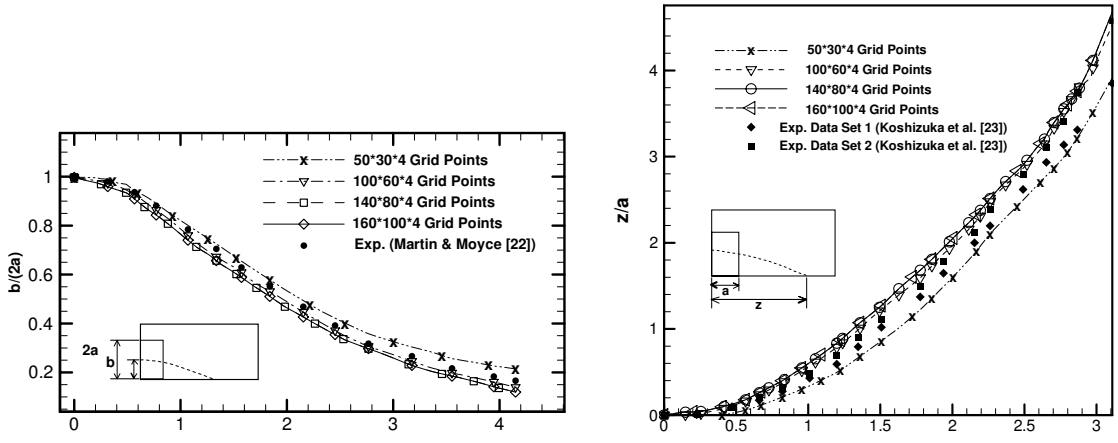


Figure 1: Schematic of dam-breaking testing case. Slip conditions have been applied to the front and back wall of the tank, non-slip conditions have been applied to the rest of walls (left, right, up and down).



(a) $t\sqrt{g/a}$ (b) $t\sqrt{2g/a}$

Figure 2: (a) The height of the collapsing water column versus time; (b) The position of leading edge versus time. For all simulations, ILU preconditioned GMRES is used to solve the linear system.

To begin with, numerical predictions for five successively finer mesh sizes with $50\times30\times4$ (Length$\times$Height$\times$Width), $60\times40\times4$, $100\times60\times4$, $140\times80\times4$ and $160\times100\times4$ grid points respectively are presented. All of these tetrahedral grids are generated from uniform spaced structured mesh. The predicted non-dimensional height of the collapsing water column at the left wall versus the non-dimensional time is shown in Fig. 2(a), and the non-dimensional positions of the leading edge for the same cases are shown in Fig. 2(b). As shown in the figures, for all of these calculations the predicted results correspond very well with the experimental data presented by Martin and Moyce [22]. The calculated results show that the leading edge moves faster when the resolution of the mesh increases. Results presented by other researchers show the same tendency (see [23]). The reason for this is the difficulty to determine the exact position of the leading edge. A thin layer shoots over the bottom and the rest of the bulk flow follows shortly behind it. The difficulty is also confirmed by Martin and Moyce in [22] who present two different sets of experimental data. To determine the overall accuracy of the proposed methods, we carried out a grid convergence study. In this study, the solution on finest-mesh ($160\times100\times4$) at $t=0.3$s is considered to be the "benchmark" solution. On all the grids the same physical time step (0.005s) is employed in order to concentrate on the spatial resolution of the method. For all the grids, the simulation time is set to 0.3s, at the end of which the infinite and k^{th} norms of the level set function errors are calculated as follows:

$$\varepsilon_N^{\infty}=\max_{i=1,M}\left|\varphi_i^N-\varphi_i^e\right|,\qquad \varepsilon_N^k=\left[\frac{1}{M}\sum_{i=1}^{M}\left|\varphi_i^N-\varphi_i^e\right|^k\right]^{\frac{1}{k}},\tag{4.1}$$

where superscript N represents the grid refinement level (50, 60, 100 and 140) and M the number of grid points satisfying the condition $|\varphi_i|\leq5\Delta h$ and Δh is the local grid size. ε_N^{∞}

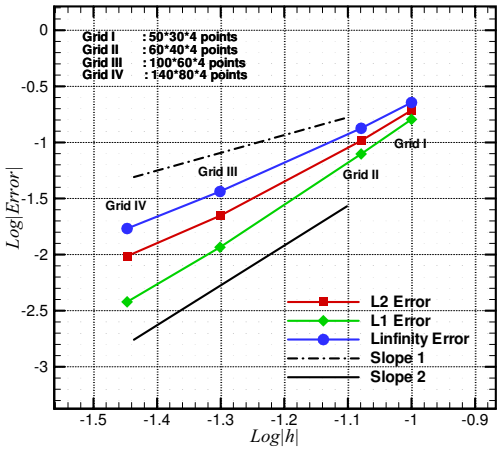


Figure 3: Convergence of the L_∞, L_1 and L_2 error norms for the single vortex flow testing case. Slope 1 and Slope 2 are the reference lines for 1-order and 2-order accuracy respectively. All of the error norms are computed within the transitional region near the zero level set.

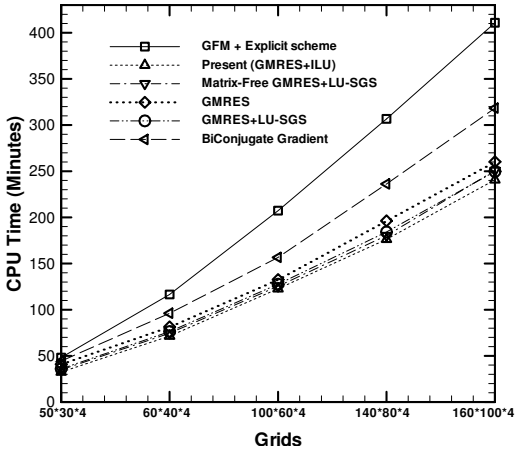


Figure 4: The comparisons of CPU time needed by different schemes. The implementation of GFM was following [26] with the difference that the incompressible and compressible solvers for the two separated phases are implemented using our own method [11].

and ε_N^k are the infinity and k^{th} error norms, φ_i^N is the level set function at node i, and φ_i^e is the "exact" level set field.

The results of the grid convergence study are summarized in Table 1 as well as Fig. 3. The graph shows the variation of the L_∞, L_1 and L_2 norms of errors with grid spacing in logarithmic coordinates. The lines with slope one and two are also given as reference.

Table 1: The computed error norms L_∞, L_1 and L_2 from Eq. (4.1) on four grids for dam breaking case.

Grids	L_∞	L_1	L_2
50×30×4	2.2675e-1	1.6053e-1	1.9300e-1
60×40×4	1.3358e-1	7.8920e-2	1.0364e-1
100×60×4	3.6529e-2	1.1632e-2	2.2163e-2
140×80×4	1.7082e-2	3.7920e-3	9.6768e-3

A conclusion that can be drawn from the results is that the mesh with 140×80×4 grid points can give accurate enough results for our purpose, and to save CPU time, this mesh will be used in following tests.

The total CPU time (in minutes) needed by current method using different linear system solvers for evolving the leading edge to a given position ($x = 5a$, right wall) on the five meshes are shown in Fig. 4. All the simulations are carried out on a PC with 4GB memory and an Intel Dual-Core CPU Q6600 running at 2.4GHz.

It is obvious that GMRES+ILU performs best in this particular case and on the finest grid (with 160×100×4 grid points) it saves up to 40% time compared to the explicit GFM.

This big amount of time saving can be easily understood by the fact that in the current implicit implementation, the multi-stage Runge-Kutta explicit time-integration is no longer needed. This means in each time step, the numerical fluxes are computed only once other than multiple times. Furthermore, due to the improved convergence performance, 10+ pseudo sub-iterations are enough to produce the same residual drop as the explicit solver usually does using 30+ iterations (as the explicit solver employs 5-stage Runge-Kutta scheme, a total of (30+)×5=150+ residual evaluations are needed). The matrix-free GMRES+LU-SGS (see [18,27]) and GMRES+LU-SGS (see [9]) have shown similar performance on this problem and they both outperform non-preconditioned GMRES, which in turn outperforms the BiConjugate Gradient method (see [8]). In this work, a fixed number of search directions is employed $(l=20)$ and a uniform convergence tolerance $\varepsilon=10^{-3}\cdot|Res(W)|_2$ has been set for all of the GMRES solvers. And we have found that all of the preconditioned GMRES solvers always converge within 3∼5 iterations while the non-preconditioned one usually needs 50+ iterations to converge.

A remarkable advantage of currently proposed free-surface capture scheme is that we are now able to make the choice to solve either the complete two-phase or only liquid phase. And the shift between the two types of computation can be done in a seamless manner by changing the deactivating interface position between 0 and 0.5 [10]. Setting this interface to a position where VOF value is less than 0.3 is not possible in our previous implementation in order to maintain numerical stability if the density ratio is high. The two-phase governing equations are now properly preconditioned and hence well-conditioned; so we can solve the complete two phases by setting the deactivating interface to $F=0$, and now the extension of velocity and pressure [10] from within the liquid phase is not needed. Two snapshots of the simulation at different instants are shown in Fig. 5. We can spot several vortices within the gas domain and this is due to the strong interaction between the two phases. Now it would be interesting if we compare the results of "single-phase" simulation and "two-phase" simulation to see how big the difference is and in this way we could verify the accuracy of our proposed "air deactivating" method. To serve this purpose, we pick up a point P in the domain as shown in Fig. 6 and monitor the time histories of (p,u,v,w) for both case and compare. The selection rule for this point is quite simple: it should be always emerged in water so that the monitored flow variables are valid all the time. The comparisons are shown in Fig. 6 and three data sets are included. As expected, the results obtained by setting the *position of deactivation interface* $F=0.01$ is more accurate than by setting it equal to 0.5 if we consider the "two-phase" simulation results as the "benchmark".

As mentioned earlier in this paper, the new preconditioning scheme will make the two-phase governing equations better conditioned and the preconditioned eigensystem is independent of density ratio. In Fig. 7, the predictions of the non-dimensional positions of the leading edge of breaking dam for three different density ratios are shown. This confirms our assertion that the proposed method is capable of solving large density ratio two-phase problems.

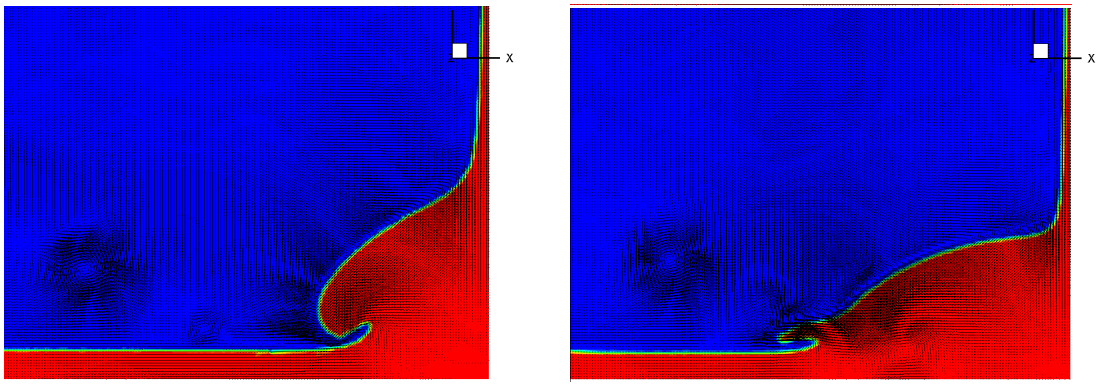


Figure 5: The velocity and VOF fields at two instants of simulation, the lower-right corner of the domain is shown. VOF contours are drawn at 15 levels equally spaced intervals between 0 and 1.

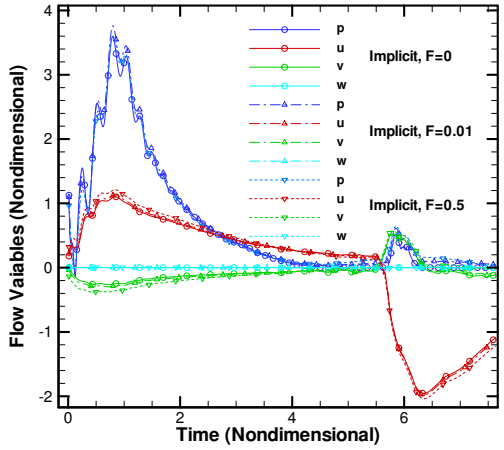


Figure 6: The comparisons of solution accuracy by setting different values of deactivation interface position ($F=0$, $F=0.01$ and $F=0.5$). The grid with $140\times80\times4$ points were used. GMRES+ILU employed.

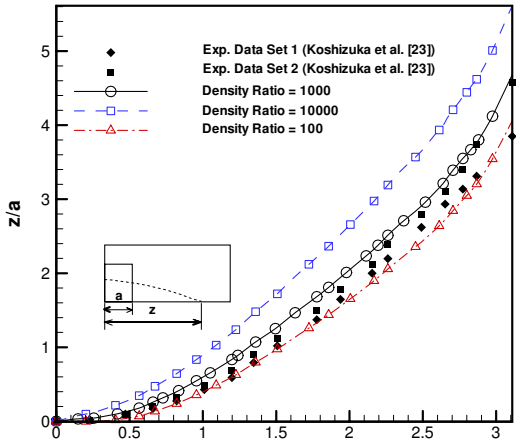


Figure 7: The predicted position of leading edge versus time. Three different density ratios are considered. Implicit GMRES+ILU scheme is employed.

4.2 Dam breaking wave interacting with an obstacle

The next selected problem is the interaction of dam breaking wave with an obstacle. The measurement data for this case is available due to the experimental work done at MAritime Research Institute Netherlands (MARIN) [24, 25]. In this experiment, a large tank of dimensions $3.22\times1.0\times1.0$m is used with an open roof. The right part of the tank is first closed by a door. Behind the door 0.55m of water is waiting to flow into the tank when the door is opened. In the left part of the tank an obstacle has been placed that represents a scale model of a container on the deck of a ship. Fig. 8 shows the schematic of computational domain. During the experiment measurements have been performed of water heights, pressures and forces. In Fig. 8 the positions of the measured quantities

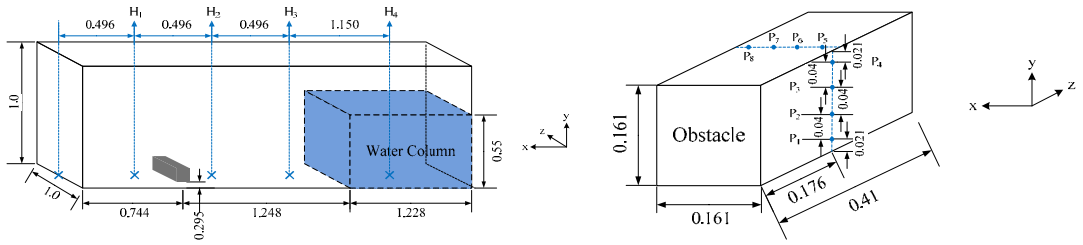


Figure 8: General description of the system for the simulation of interaction between dam breaking wave and an obstacle.

are shown as well. Four vertical height probes have been used ($H_1 \sim H_4$); one in the reservoir and the other three in the tank. The obstacle was covered by eight pressure sensors ($p_1 \sim p_8$), four on the front of the box and four on the top. As initial configuration of the current simulation, the water column is at rest. When the simulation is started, due to gravity the water starts to flow into the empty part of the tank. A fine grid of 1,008,857 grid points and 5,882,357 tetrahedral elements has been used with some local refinements near the obstacle and the walls of the tank. The simulation is continued for 6s with an automatically adapted time step using maximum CFL-numbers 1.0 resulting in the time step size of the order of 0.005s. In Fig. 9 several snapshots of the early stages of the simulation are shown. Good agreements can be seen between the snapshots of simulation and experiment (snapshots of the experiment can be found in [24] and [25]).

In Fig. 10 time histories of the predicted water depth at two locations are shown (we choose the same locations as [24] for ease of comparison): in the reservoir (H_4) and in the tank just in front of the box (H_2). Good agreement at location H_4 can be identified from the left picture of Fig. 10. After the wave front has been reflected from the left wall, the fluid height at probe H_2 is the largest (about 1.8s). This fact is confirmed by the experiment data as well as the numerical results presented in [24]. As can be seen from the right picture of Fig. 10, the agreement at location H_2 before 1.8s is extremely good until the water has returned from the back wall. After that some differences occur, but the global flow behaviour is reasonably good. The wave front flows back to the reservoir, where it hits the right wall and turns over again after about 4s. A detectable difference at probe H_2 between simulation and experiment at this point (4s) is that in the experiment there is a notable increase of water depth followed by a quick drop, while this change is missing in the simulation. However, the moment that this second wave meets the height probe at H_2 again (after about 5s) is almost exactly the same in simulation and experiment.

The instant when the wave hits the obstacle is extremely well captured by the simulation as can be seen from Fig. 11, where the pressures at point P_1, P_3, P_5 and P_7 (see Fig. 8), are shown. The magnitude of the impact pressure is very close for simulation and experiment at pressure point P_1, but is slightly under predicted by the simulation at point P_3. This disagreement can be diminished further by refining the mesh near the

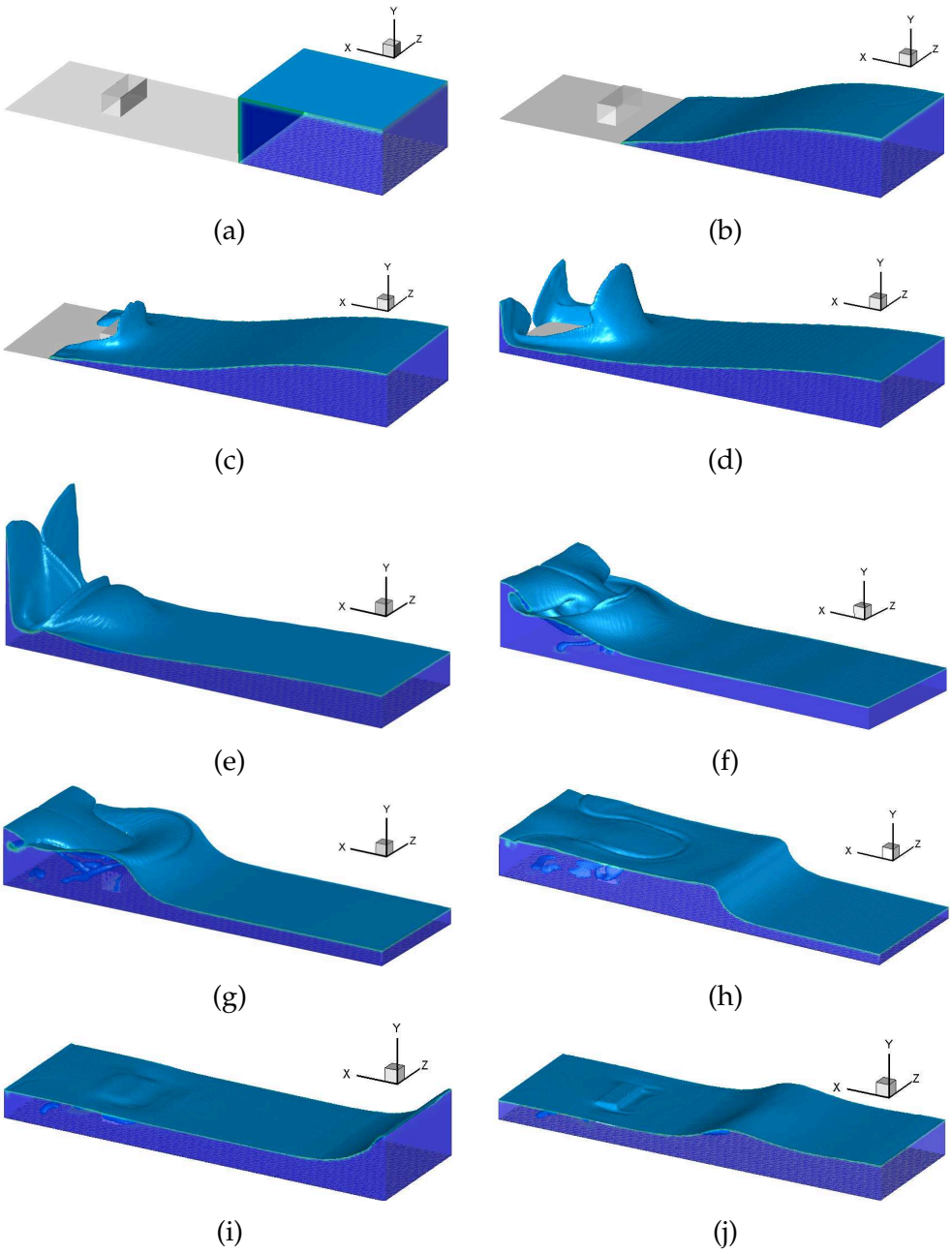


Figure 9: Snapshots of predicted dam breaking wave interacting with obstacle in early stages. ISO surface by F=0.5. Implicit scheme, GMRES+ILU on mesh with 1,008,857 grid points and 5,882,357 tetrahedral elements.

block. In the bottom graphs of Fig. 11, where the time histories of pressure at the top of the box (P_5 and P_7) are shown, and a clear difference between simulation and experiment occurs after about 1.3s. There is a wiggle in the simulation with approximate duration of

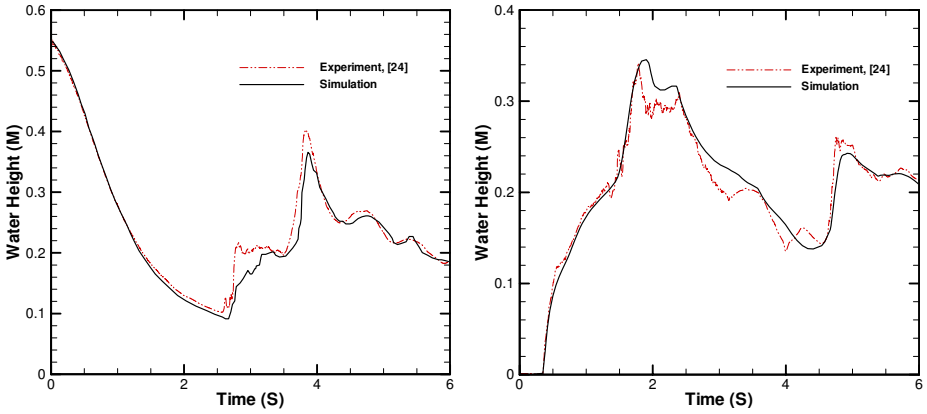


Figure 10: Predicted water depth in the reservoir H_4 (left) and the tank H_2 (right).

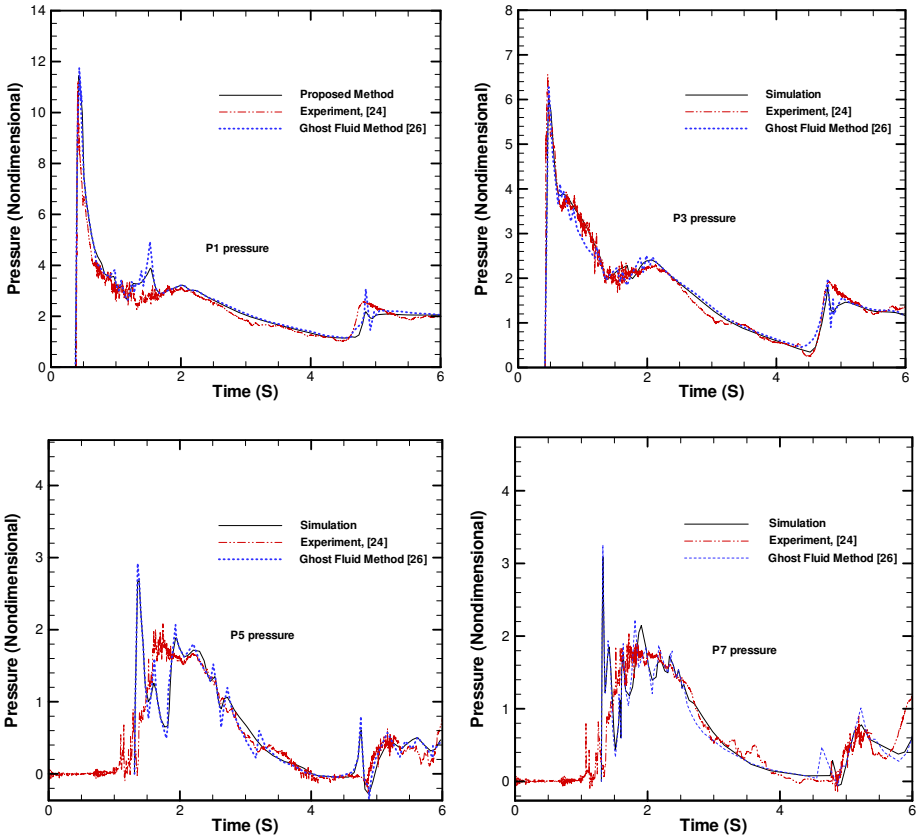


Figure 11: Predicted and measured pressure time histories at P_1 (upper left), P_3 (upper right), P_5 (bottom left) and P_7 (bottom right). Results by GFM (see [26]) are also presented for comparison. The pressure was nondimensionalized by dividing $\rho_{ref}U^2_{\max}$, where ρ_{ref} is the density of water and $U_{\max}$ the maximum velocity in the field.

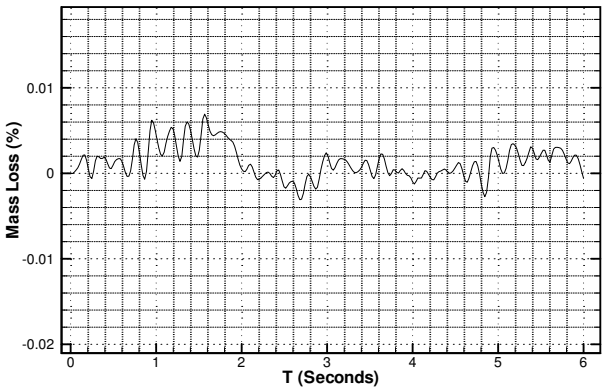


Figure 12: Mass loss time histories for the test case of Dam Breaking Wave Interacting with an Obstacle, the proposed method can keep the mass loss as low as 0.7%.

0.55s, which is not present in the experiment (or of much smaller magnitude). Attempts to improve the result by locally refining the grid or adjusting the time step size were not successful. This difficulty has also been reported in [24]. The moment the return wave hits the obstacle again (at about 4.72s) is again visible in the graphs and agrees well with the experiment data. The numerical results by GFM [26] are also provided in Fig. 11 for comparison purpose. As can be seen from the figures, both methods produce similar results and can predict the pressure to a reasonably accurate level, while the result from our current method exhibits a smoother impact and we would claim a more physically realistic representation.

Finally, the mass loss history during the whole simulation is shown in Fig. 12. As is evident the proposed method preserves the water mass very accurately (mass loss$<$ 0.7%).

4.3 Single bubble rising in a vertical column

This section considers isothermal, incompressible flows of immiscible fluids where the conservation of momentum and mass is described by the Navier-Stokes equations (2.1). Unlike the previous dam breaking problem, surface tension effects are important and may not be neglected. The initial configuration, shown in Fig. 13, consists of a circular bubble of radius $r_0 = 0.25$ centered at $[1.25,0.5]$ in a $[2.5\times 3.5]$ rectangular domain [39]. The density of the bubble is smaller than that of the surrounding fluid $(\rho_2 < \rho_1)$ so that it will rise upwards. The setup of boundary conditions is also described in the figure. The benchmarks are restricted to two dimensions since both computational complexity and time is greatly reduced. This test case serves as a good example exhibiting the improvements made by current model against the single phase model described in [10] when solving small scale free surface problems where the effects of the fluid with smaller density are important.

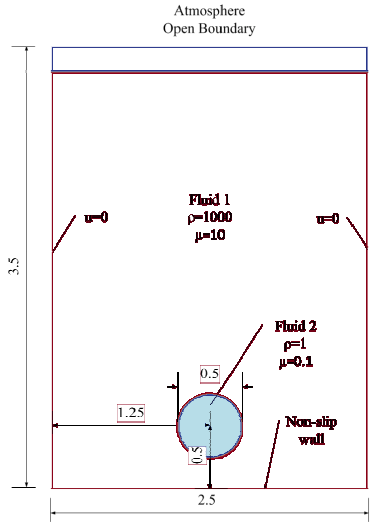


Figure 13: Schematic view of initial configuration and boundary conditions.

Table 2 lists the fluid and physical parameters. The evolution of the bubble is tracked for multiple times on 4 different meshes. And the time step size is fixed to $\Delta t = h/10$, and h is the mean grid size.

Table 2: Physical parameters and dimensionless numbers defining the benchmark test case.

$\rho_1(\mathrm{kg/m^3})$	$\rho_1(\mathrm{kg/m^3})$	μ_1 (kg/m·s)	μ_2 (kg/m·s)	$G(\mathrm{m/s^2})$	$\sigma(\mathrm{kg/s^2})$	Re	E_o	ρ_1/ρ_2	μ_1/μ_2
1000	1	10	0.1	0.981	1.96	35	125	1000	100

This test case models a rising bubble with $Re = 35$, $Eo = 125$, and with large density and viscosity ratios (1000 and 100). This bubble lies somewhere between the skirted and dimpled ellipsoidal-cap regimes indicating that break up can possibly occur [35], which will present additional challenges to the different interface tracking algorithms. The dimensionless Reynolds and Eötvös numbers are here specifically defined to relate to bubbles as

$$Re = \frac{\rho_1 \sqrt{G}(2r_0)^{\frac{3}{2}}}{\mu_1}, \qquad E_0 = \frac{4\rho_1 G r_0^2}{\sigma},$$

where the subscript 1 refers to the heavier fluid and 2 to the lighter fluid in the bubble. Moreover, r_0 is the initial radius of the bubble, σ the surface tension coefficient and G the gravitational constant.

Visual comparison of the bubble shape is one common way to compare simulation results. And besides, the following two quantities, which will be used to assist in describing the temporal evolution of the bubbles quantitatively, are introduced.

Point Quantities. Positions of various points can be used to track the translation of

bubbles. It is common to use the center of mass [36], defined by

$$X_c = (x_c, y_c) = \frac{\int_\Omega x dx}{\int_\Omega 1 dx}, \tag{4.2}$$

where Ω denotes the region that the bubble occupies. Other points could be the absolute top or bottom of a bubble [36].

Rise Velocity. The mean velocity with which a bubble is rising within the column is a particularly interesting quantity since it not only measures how the interface tracking algorithm behaves but also the quality of the overall solution. It is defined as

$$U_c = \frac{\int_\Omega u dx}{\int_\Omega 1 dx}. \tag{4.3}$$

The temporal evolution of the computed benchmark quantities can be measured against suitable reference solutions to establish the following relative error norms

$$l_1 \text{ error}: \quad \|e\|_1 = \frac{\sum_{t=0}^{T} |q_{t,ref} - q_t|}{\sum_{t=0}^{T} |q_{t,ref}|}, \tag{4.4a}$$

$$l_2 \text{ error}: \quad \|e\|_2 = \left(\frac{\sum_{t=0}^{T} |q_{t,ref} - q_t|^2}{\sum_{t=0}^{T} |q_{t,ref}|^2} \right)^{\frac{1}{2}}, \tag{4.4b}$$

$$l_\infty \text{ error}: \quad \|e\|_\infty = \frac{\max_t |q_{t,ref} - q_t|}{\max_t |q_{t,ref}|}, \tag{4.4c}$$

where q_t is the temporal evolution of quantity q. The solution computed on the finest grid is usually taken as a reference solution $q_{t,ref}$. Interpolation should be appropriately applied if there are more time steps for the reference solution than the solutions q_t for which the error norms should be computed. With the relative errors established and CPU times measured it is then easy to see how much effort is required to acquire a certain accuracy level. Additionally, convergence rates for the quantities can also be computed as

$$ROC_l = \log_{10}\left(\frac{\|e_{l-1}\|}{\|e_l\|} \right) \Big/ \log_{10}\left(\frac{h_{l-1}}{h_l} \right), \tag{4.5}$$

where l is the grid level. In this study, totally 4 level of grids with different resolutions were used to perform the detailed grid convergence studies. The average grid sizes for the grids are 1/50, 1/100, 1/200 and 1/400 respectively.

Benchmarking and validation efforts have been conducted with the aim of producing grid independent reference solutions. The reason why this test case was proposed is because extensive initial studies have already been carried out by the many other research groups, FreeLIFE from EPFL for example [37, 38]. As shall be seen in the following, the current study found very good agreement for the proposed test case.

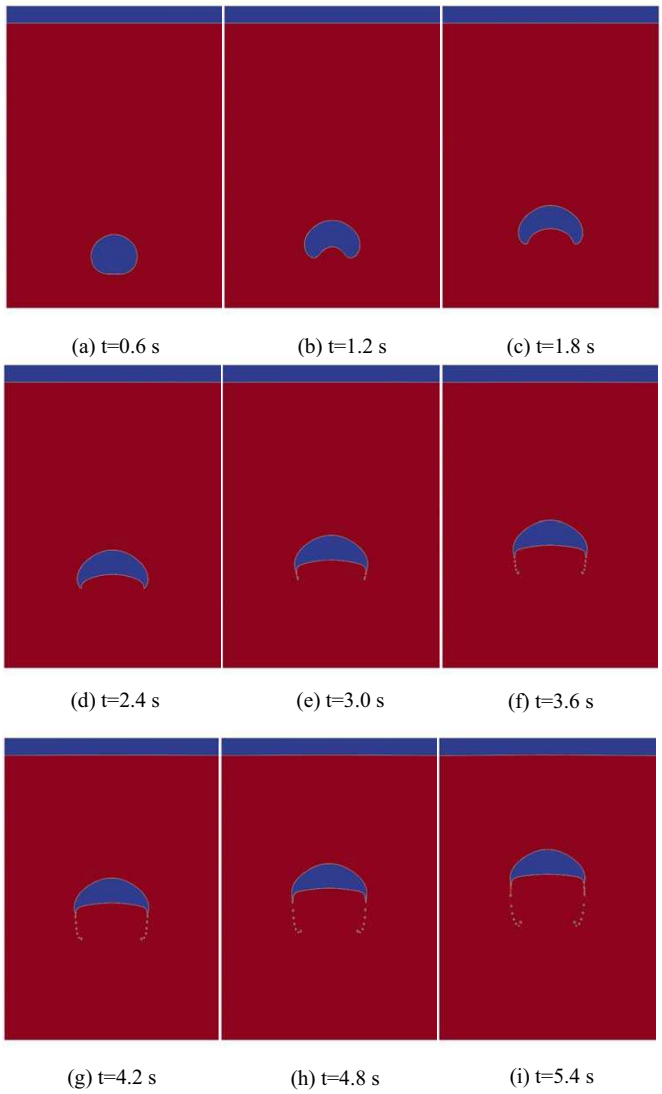
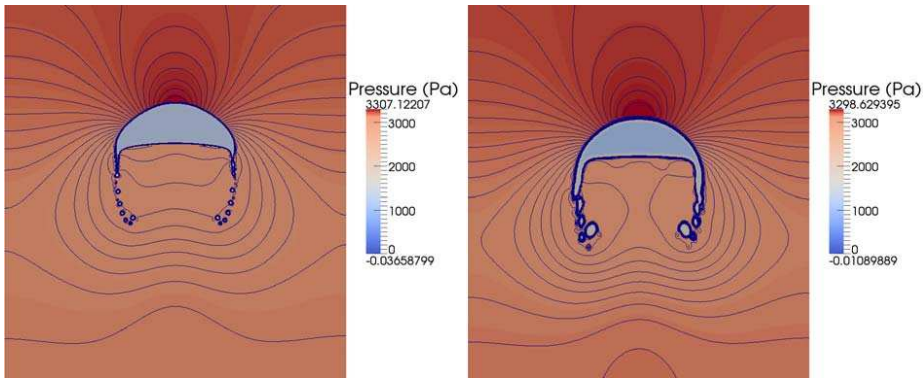


Figure 14: Time evolution of the single rising bubble. The bubble shape is contoured by VOF number $F=0.5$.

Fig. 14 shows the snapshots of the time evolution of the bubble (computed on $h=1/160$ grid). As can be seen, the surface tension causes the initial round bubble to assume a convex shape and develop thin filaments which eventually break off, as is evident from Fig. 14. After the break up small satellite droplets trail the bulk of the main bubble, which eventually assumes the shape of a dimpled cap.

The bubble shapes at the final simulation time ($t=5.4$), computed by both the current proposed model and the explicit model reported in [10], are shown in Fig. 15. First of all one can see that the simulation by the single phase explicit model produced a rather un-



(a) (b)

Figure 15: Comparison of the computed bubble shapes at $t=5.4$s, contoured by pressure. (a) final bubble shape computed by current proposed model; (b) final bubble shape computed by the single phase explicit model reported in [10].

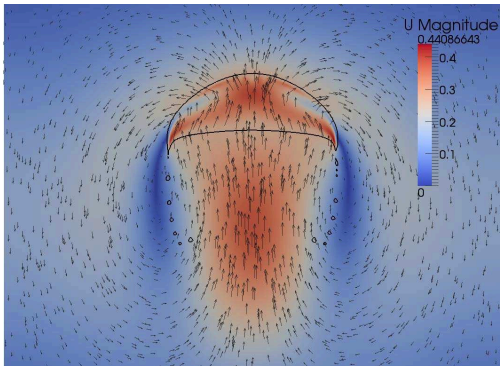


Figure 16: Instant shape of the single rising bubble at $t=5.4$s, contoured by VOF number $F=0.5$. Figure is colored by velocity magnitude. Velocity vectors are also plotted.

physical break up behaviour, producing sharp edged trailing filaments (Fig. 15(b)). The shape computed by the current model did not have these filaments and seemed to converge for the main bulk bubble. The two models gave quite different results in predicting the bubble evolution, and as can be seen from what follows, the result by the current proposed model is more accurate. The reason is actually obvious viewing the fact that, in the single phase model, the jump condition is applied by employing the extrapolated flow field since flow field in fluid 2 is not solved but estimated. However in the current model, the flow fields of both fluids are calculated and thus the jump condition is treated in a natural and accurate manner.

Fig. 16 shows the distribution of velocity magnitude at $t=5.4$s. The vectors by the velocity components are also plotted so one may have a clear picture of the flow field, including the inside domain of the bubble.

Table 3 shows the simulation statistics and timings for this test case. The Using the proposed implicit model, the CPU times have reduced significantly, as shown in Table 3. This is mostly due to the elimination of the need of the time consuming Runge-Kutta sub-iterations in each time step.

Table 3: Simulation statistics for single rising bubble. All test runs stopped at $t=5.4$s.

Grid size	Num of elements	Num of nodes	CPU time (unit in second)				CPU time reduced in percentage
			Current model	CPUs used	Single phase model [10]	CPUs used	
1/50	150,106	50,188	535	8	661	8	19.06%
1/100	755,855	225,973	2,762	16	3,489	16	20.84%
1/200	6,357,378	1,460,412	9,128	32	14,511	32	37.10%
1/400	51,216,648	12,194,440	24,538	128	33,996	128	27.82%

A quantitative convergence analysis has been performed computing the relative error norms for the center of mass and rise velocity, together with the estimated rates of convergence (ROC, defined in Eq. (4.5)). Both the analysis results of current model and single phase model reported in [10] are listed in Table 4. Here, the reference solution is taken as the solution from the computation on the finest grid ($h=1/400$). It is evident that all quantities converge with a faster convergence order in current model than in the single phase model. This is easy to understand as the flow field extrapolation (a key component in single phase model) can introduce in considerable amount of numerical oscillations which in turn cause the fluctuations in the computed flow variables, as can be seen from the next two figures.

Table 4: Computed relative error norms and convergence orders for single bubble rising by both models. Current model designated as (a) and the single phase model reported in [10] designated as (b).

h	$\|e\|_1$		$\mathbf{ROC}_1$		$\|e\|_2$		$\mathbf{ROC}_2$		$\|e\|_\infty$		$\mathbf{ROC}_\infty$	
	(a)	(b)	(a)	(b)	(a)	(b)	(a)	(b)	(a)	(b)	(a)	(b)
Center of mass												
1/50	3.22e-03	3.67e-03			3.62e-03	4.28e-03			4.48e-03	5.56e-03		
1/100	7.61e-04	1.02e-03	2.08	1.84	8.32e-04	1.52e-03	2.12	1.49	1.17e-03	2.24e-03	1.94	1.31
1/200	1.75e-04	2.62e-04	2.12	1.96	2.21e-04	5.14e-04	1.91	1.56	2.97e-04	7.76e-04	1.98	1.53
Rise velocity												
1/50	1.11e-02	1.39e-02			1.43e-02	3.13e-02			1.79e-02	4.49e-02		
1/100	2.35e-03	4.25e-03	2.24	1.71	3.47e-03	9.27e-03	2.04	1.76	5.18e-03	1.75e-02	1.79	1.35
1/200	5.73e-04	1.13e-03	2.04	1.91	8.75e-04	3.32e-03	1.99	1.48	1.41e-03	8.14e-03	1.88	1.10

The time evolutions of the center of mass and mean rise velocity can be seen in Figs. 17 and 18, respectively. Both these quantities computed by current model seem to agree well with that of the benchmark code although the curve from the simulation using single phase model deviates somewhat from the other two. According to the current model, the center of mass reaches a height of 1.1243 at the time of $t=3.0$s on the finest grid. Two velocity maximum can be easily identified in Fig. 18. A very good agreement can be seen for the curves describing the rise velocity up until the first maximum, occurring at $t=0.7383$s with a magnitude of 0.2511 on the finest grid. And this is due to the fact

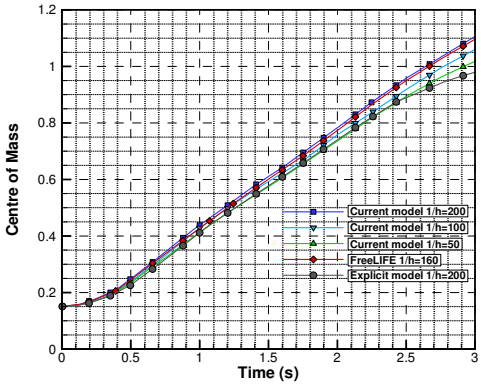


Figure 17: Time history of the mass centre. Results by FreeLIFE [37, 38] and previously reported explicit model [10] are also presented for comparisons.

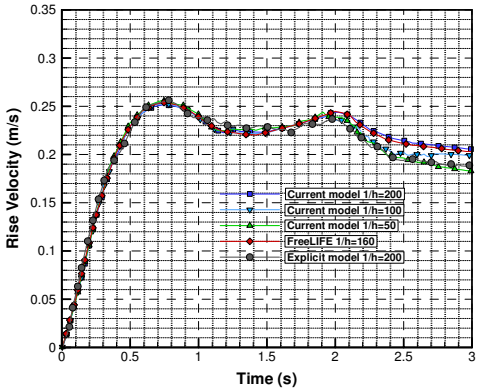


Figure 18: Rise velocity against time. Results by FreeLIFE [37, 38] and previously reported explicit model [10] are also presented for comparisons.

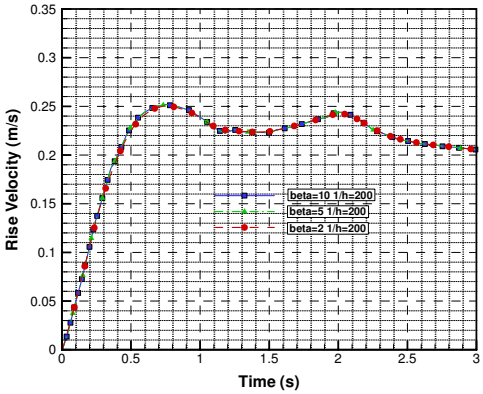


Figure 19: Sensitivity study of the ACM coefficient β.

that by now the changes in the shape of the bubble is small. But from then on the curve corresponding to the simulations on the coarse grids, as well as using the single phase model, starts to show a somewhat irregular and oscillatory behaviour, most likely due to some oscillations in the velocity field close to the interface. On coarse grids, these oscillations are expected because of the low grid resolution. But for the single phase model, the oscillations still present even on the finest grid, probably because of the extrapolation procedure, as explained earlier.

It should be noted that in this case almost all the model parameters $(\Delta t, \Delta\tau, Re, \sigma)$ are given as standards to both provide the research community with reference data and also elicit more participation from additional groups [39].

The sensitivity of the ACM coefficient β has been evaluated and confirmed to be negligible. Please have a look of the Fig. 19, where the effects of changing the values of β to the simulation results are presented.

5 Conclusions

The majority of two-phase flow approaches in the literature are either based on an explicit formulation, or semi-implicit (projection method) formulation, but rarely fully implicit (see [32–34]). In this paper we have presented the development of a 3D preconditioned implicit two-phase free-surface capture solver which improves upon our recently reported method in [10]. The core of present method is the coupled Level Set and VOF method which combines the strengths of both the level set and volume of fluid methods. To enhance robustness and improve efficiency, the time integration scheme has been changed from an explicit multi-stage Runge-Kutta integration to a fully implicit Euler scheme. In addition the VOF equation is now evolved by a second order Crank-Nicolson implicit scheme to reduce numerical diffusion. The preconditioned restarted GMRES is then employed to solve the resulting linear system. The numerical simulation of the dam breaking (and its interaction with an obstacle) is presented to show that both the model accuracy and efficiency is improved. Being able to set the values of the *deactivation interface position* to a smaller value, the result of "single phase" simulation by deactivating the air phase is now closer to that of two-phase simulation, which is more accurate because in this situation the air deactivation does not occur and thus the accuracy is not affected by the extrapolation scheme introduced in the free-surface boundary treatment. Thus, in our new method, the air deactivation technique is employed solely to reduce CPU time. As shown in Section 4, the proposed method can predict the free surface evolution more accurately while consuming much less CPU time. The benefit of preconditioning is evident as the solver is now virtually independent of the density ratio. Validation of a dam break wave interacting with an obstacle showed that our proposed method is capable of simulating a free surface flow with complex interface changes and high curvatures accurately and efficiently. The benchmark study of the proposed model on a single bubble rising in a column further confirms its overall improvements over the explicit counter-

part reported in [10], both in terms of accuracy and efficiency for two phase problems with big density ratio.

The method will be developed further in the coming years by extending it towards handling floating bodies and violent overtopping of sea walls. An adaptive re-mesh algorithm is also under development which we believe can greatly reduce the computational workload.

Appendix

Of interest, in the construction and analysis of a scheme to discretize and solve Eqs. (2.1)-(2.6), is its inviscid eigensystem. In particular, the eigenvalues and eigenvectors of matrix $\hat{A}_c$ are required, where

$$\hat{A}_c=\Gamma^{-1}A_c=\Gamma^{-1}\frac{\partial\vec{F}_c}{\partial W}=\Gamma^{-1}\begin{bmatrix}0 & n_x & n_y & n_z & 0\\ \rho^{-1}n_x & n_xu+U & n_yu & n_zu & 0\\ \rho^{-1}n_y & n_xv & n_yv+U & n_zv & 0\\ \rho^{-1}n_z & n_xw & n_yw & n_zw+U & 0\\ 0 & n_x\phi & n_y\phi & n_z\phi & U\end{bmatrix}. \tag{A.1}$$

Note that

$$\Gamma=\begin{bmatrix}\frac{1}{\rho\beta} & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ \frac{\phi}{\rho\beta} & 0 & 0 & 0 & 1\end{bmatrix},\qquad \Gamma^{-1}=\begin{bmatrix}\rho\beta & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ -\phi & 0 & 0 & 0 & 1\end{bmatrix}. \tag{A.2}$$

Thus we have

$$\hat{A}_c=\Gamma^{-1}A_c=\begin{bmatrix}0 & \rho\beta n_x & \rho\beta n_y & \rho\beta n_z & 0\\ \rho^{-1}n_x & un_x+U & un_y & un_z & 0\\ \rho^{-1}n_y & vn_x & vn_y+U & vn_zv & 0\\ \rho^{-1}n_z & wn_x & wn_y & wn_z+U & 0\\ 0 & 0 & 0 & 0 & U\end{bmatrix}. \tag{A.3}$$

Its eigensystem reads:

$$\hat{A}_c=\hat{T}\cdot\hat{\Lambda}\cdot\hat{T}^{-1}, \tag{A.4}$$

where

$$\hat{T}=\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ \frac{(u+cn_x)\beta+\lambda_1 uU}{c^2\rho\beta} & \frac{(u-cn_x)\beta+\lambda_2 uU}{c^2\rho\beta} & -\frac{n_z}{n_x} & 0 & -\frac{n_y}{n_x} \\ \frac{(v+cn_y)\beta+\lambda_1 vU}{c^2\rho\beta} & \frac{(v-cn_y)\beta+\lambda_2 vU}{c^2\rho\beta} & 0 & 0 & 1 \\ \frac{(w+cn_z)\beta+\lambda_1 wU}{c^2\rho\beta} & \frac{(w-cn_z)\beta+\lambda_2 wU}{c^2\rho\beta} & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \tag{A.5}$$

$$\hat{T}^{-1}=\frac{1}{c}\begin{bmatrix} -\frac{\lambda_2}{2} & \frac{\rho\beta n_x}{2} & \frac{\rho\beta n_y}{2} & \frac{\rho\beta n_z}{2} & 0 \\ \frac{\lambda_1}{2} & -\frac{\rho\beta n_x}{2} & -\frac{\rho\beta n_y}{2} & -\frac{\rho\beta n_z}{2} & 0 \\ \frac{(Un_z-w)}{\rho c} & -\frac{(wU+\beta n_z)n_x}{c} & -\frac{(wU+\beta n_z)n_y}{c} & \frac{(c^2-(wU+\beta n_z)n_z)}{c} & 0 \\ 0 & 0 & 0 & 0 & c \\ \frac{(Un_y-v)}{\rho c} & -\frac{(vU+\beta n_y)n_x}{c} & \frac{(c^2-(vU+\beta n_y)n_y)}{c} & -\frac{(vU+\beta n_y)n_z}{c} & 0 \end{bmatrix}, \tag{A.6}$$

$$\hat{\Lambda}=\begin{bmatrix} \lambda_1 & 0 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 & 0 \\ 0 & 0 & 0 & \lambda_4 & 0 \\ 0 & 0 & 0 & 0 & \lambda_5 \end{bmatrix}=\begin{bmatrix} U+c & 0 & 0 & 0 & 0 \\ 0 & U-c & 0 & 0 & 0 \\ 0 & 0 & U & 0 & 0 \\ 0 & 0 & 0 & U & 0 \\ 0 & 0 & 0 & 0 & U \end{bmatrix}, \tag{A.7}$$

$$U=un_x+vn_y+wn_z, \qquad c=\sqrt{U^2+\beta}. \tag{A.8}$$

The viscous Jacobian presented in Eqs. (2.25)-(2.27) is derived by taking into account the method we employed in discretizing the derivatives of solution variables. Alternatively, they can be expressed as

$$\frac{\partial F_v^3}{\partial W}=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2\frac{\partial}{\partial x} & 0 & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\frac{\partial F_v^3}{\partial W}=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 \\ 0 & 0 & 2\frac{\partial}{\partial y} & 0 & 0 \\ 0 & 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\frac{\partial F_v^3}{\partial W}=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & 0 & 2\frac{\partial}{\partial z} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{A.9}$$

$$A_v=\frac{\partial F_v^3}{\partial W}\cdot n_x+\frac{\partial F_v^3}{\partial W}\cdot n_y+\frac{\partial F_v^3}{\partial W}\cdot n_z$$

$$=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2\frac{\partial}{\partial x}n_x+\frac{\partial}{\partial y}n_y+\frac{\partial}{\partial z}n_z & \frac{\partial}{\partial x}n_y & \frac{\partial}{\partial x}n_z & 0 \\ 0 & \frac{\partial}{\partial y}n_x & \frac{\partial}{\partial x}n_x+2\frac{\partial}{\partial y}n_y+\frac{\partial}{\partial z}n_z & \frac{\partial}{\partial y}n_z & 0 \\ 0 & \frac{\partial}{\partial z}n_x & \frac{\partial}{\partial z}n_y & \frac{\partial}{\partial x}n_x+\frac{\partial}{\partial y}n_y+2\frac{\partial}{\partial z}n_z & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{A.10}$$

$$
\begin{aligned}
F_v^{m+1} &= F_v^m + \frac{\partial F_v^m}{\partial W}\Delta W \\
&= F_v^m + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2\frac{\partial}{\partial x}n_x+\frac{\partial}{\partial y}n_y+\frac{\partial}{\partial z}n_z & \frac{\partial}{\partial x}n_y & \frac{\partial}{\partial x}n_z & 0 \\ 0 & \frac{\partial}{\partial y}n_x & \frac{\partial}{\partial x}n_x+2\frac{\partial}{\partial y}n_y+\frac{\partial}{\partial z}n_z & \frac{\partial}{\partial y}n_z & 0 \\ 0 & \frac{\partial}{\partial z}n_x & \frac{\partial}{\partial z}n_y & \frac{\partial}{\partial x}n_x+\frac{\partial}{\partial y}n_y+2\frac{\partial}{\partial z}n_z & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta u \\ \Delta v \\ \Delta w \\ \Delta \phi \end{bmatrix} \\
&= F_v^m + \underbrace{\begin{bmatrix} 0 \\ 2\frac{\partial \Delta u}{\partial x}n_x+\frac{\partial \Delta u}{\partial y}n_y+\frac{\partial \Delta u}{\partial z}n_z+\frac{\partial \Delta v}{\partial x}n_y+\frac{\partial \Delta w}{\partial x}n_z \\ \frac{\partial \Delta u}{\partial y}n_x+\frac{\partial \Delta v}{\partial x}n_x+2\frac{\partial \Delta v}{\partial y}n_y+\frac{\partial \Delta v}{\partial z}n_z+\frac{\partial \Delta w}{\partial y}n_z \\ \frac{\partial \Delta u}{\partial z}n_x+\frac{\partial \Delta v}{\partial z}n_y+\frac{\partial \Delta w}{\partial x}n_x+\frac{\partial \Delta w}{\partial y}n_y+2\frac{\partial \Delta w}{\partial z}n_z \\ 0 \end{bmatrix}}_{\text{This part will be moved to the LHS of the discretized equation}}.
\end{aligned} \tag{A.11}
$$

Acknowledgments

This research was supported by the Flood Risk from Extreme Events (FREE) Programme of the UK Natural Environment Research Council (NERC) (NE/E0002129/1), coordinated and monitored by Professor Chris Collier and Paul Hardaker. We thank Dr. Zhengyi Wang and Dr. Qun Zhao for helpful discussions. The numerical calculations have been carried out on the HPC facility at the University of Plymouth.

References

[1] R. F. Kunza and D. A. Boger et al., A preconditioned Navier-Stokes method for two-phase Flows with application to cavitation prediction, Comput. Fluids, 29 (2000), 849–875.
[2] T. J. Barth, Analysis of implicit local linearization techniques for upwind the TVD algorithms, Twenty-fifth Aerospace Sciences Meeting, January 1987, AIAA Paper 87-0595.
[3] Y. Saad and M. H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput., 7(3) (1986), 856–869.
[4] V. Venkatakrishnan and D. J. Mavriplis, Implicit solvers for unstructured meshes, J. Comput. Phys., 105 (1993), 83–91.
[5] D. L. Whitaker, Three-dimensional unstructured grid Euler computations using a fully-implicit, upwind method, AIAA Paper 93-3337, 1993.
[6] H. Luo, J. D. Baum, R. Löhner and J. Cabello, Implicit schemes and boundary conditions for compressible flows on unstructured meshes, AIAA Paper 94-0816, 1994.
[7] T. J. Barth and S. W. Linton, An unstructured mesh Newton solver for compressible fluid flow and its parallel implementation, AIAA Paper 95-0221, 1995.
[8] E. H. Cuthill and J. M. McKee, Reducing the bandwidth of sparse symmetric matrices, in: Proceedings of the 24th National Conference of the Association for Computing Machinery, 1969, 157–172.

[9] A. Nejat and C. Ollivier-Gooch, Effect of discretization order on preconditioning and convergence of a high-order unstructured Newton-GMRES solver for the Euler equations, J. Comput. Phys., 227 (2008), 2366–2386.
[10] X. Lv, Q. P. Zou, Y. Zhao and D. E. Reeve, A novel coupled level set and volume of fluid method for sharp interface capturing on 3D tetrahedral grids, J. Comput. Phys., doi:10.1016/j.jcp.2009.12.005, 2009.
[11] Y. Zhao and C. H. Tai, Higher-order characteristics-based methods for incompressible flow computation on unstructured grids, AIAA J., 39(7) (2001), 1280–1287.
[12] Y. Zhao, H. H. Tan and B. L. Zhang, A high-resolution characteristics-based implicit dual time-stepping VOF method for free surface flow simulation on unstructured grids, J. Comput. Phys., 183 (2002), 233–273.
[13] W. G. Price and Y. G. Chen, A simulation of free surface waves for incompressible two-phase flows using a curvilinear level set formulation, Int. J. Numer. Methods Fluids, 51 (2006), 305–330.
[14] O. Ubbink and R. I. Issa, A method for capturing sharp fluid interfaces on arbitrary meshes, J. Comput. Phys., 153 (1999), 26–50.
[15] B. P. Leonard, The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection, Comput. Meth. Appl. Mech. Eng., 88 (1991), 17–74.
[16] A. Jameson and D. J. Mavriplis, Finite volume solution of the two-dimensional Euler equations on a regular triangular mesh, AIAA J., 24 (1986), 611.
[17] M.-S. Liou and B. van Leer, Choice of implicit and explicit operators for the upwind differencing method, 26th Aerospace Sciences Meeting, Reno, Nevada, Jan 11-14, 1988.
[18] H. Luo, J. D. Baum and R. Löhner, A fast, matrix-free implicit method for compressible flows on unstructured meshes, J. Comput. Phys., 146 (1998), 664–690.
[19] B. D. Nichols, C. W. Hirt and R. S. Hotchkiss, SOLA-VOF: A solution algorithm for transient fluid flow with multiple free boundaries, Tech. Rep., LA-8355, Los Alamos National Laboratory, 1980.
[20] C. W. Hirt and B. D. Nichols, Volume of fluid method for the dynamics of free boundaries, J. Comput. Phys., 39 (1981), 201–225.
[21] J. V. Soulis, Computation of two-dimensional dam-break flood flows, Int. J. Numer. Methods Fluids, 14 (1992), 631–664.
[22] J. C. Martin and W. J. Moyce, An experimental study of the collapse of liquid columns on a rigid horizontal plane, Philos. Trans. Roy. Soc. London, A244 (1952), 312–324.
[23] S. Koshizuka, H. Tamako and Y. Oka, A particle method for incompressible viscous flow with fluid fragmentation, Comput. Fluid Dyn. J., 4(1) (1995), 29–46.
[24] K. M. T. Kleefsman, G. Fekken, A. E. P Veldman, B. Iwanowski and B. Buchner, A volume-of-fluid based simulation method for wave impact problems. J. Comput. Phys., 206 (2005), 363–393.
[25] SPH European Research Interest Community website, `http://wiki.manchester.ac.uk/spheric/index.php/test2`.
[26] R. Caiden, R. P. Fedkiw and C. Anderson, A numerical method for two-phase flow consisting of separate compressible and incompressible regions, J. Comput. Phys., 166(1) (2001), 1–27.
[27] D. A. Knoll and D. E. Keyes, Jacobian-free Newton-Krylov methods a survey of approaches and applications, J. Comput. Phys., 193 (2004), 357–397.
[28] R. P. Fedkiw, T. Aslam, B. Merriman and S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys., 152(2) (1999),

457–492.
[29] T. G. Liu, B. C. Khoo and C. W. Wang, The ghost fluid method for compressible gas-water simulation, J. Comput. Phys., 204 (2005), 193–221.
[30] X.-D. Liu, R. P. Fedkiw and M. Kang, A boundary condition capturing method for Poisson's equation on irregular domains, J. Comput. Phys., 160 (2000), 151–178.
[31] M. Kang, R. P. Fedkiw, and X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, J. Sci. Comput., 15 (2000), 323–360.
[32] Z.-Y. Wang, Q.-P. Zou and D. E. Reeve, Simulation of spilling breaking waves using a two phase flow CFD model, Comput. Fluids, doi:10.1016/j.compfluid.2009.06.006, 2009.
[33] Y.-L. Zhang, Q.-P. Zou and D. M. Greaves, Numerical simulation of free-surface flow using the level-set method with global mass correction, Int. J. Numer. Methods Fluids, doi: 10.1002/fld.2090, 2009.
[34] D. M. Liu and P. Z. Lin, A numerical study of three-dimensional liquid sloshing in tanks, J. Comput. Phys., 227(8) (2008), 3921–3939.
[35] R. Clift, J. R. Grace and M. E. Weber, Bubbles, Drops and Particles, Academic Press: New York, 1978.
[36] L. Chen, S. V. Garimella, J. A. Reizes and E. Leonardi, The development of a bubble rising in a viscous fluid, J. Fluid Mech., 387 (1999), 61–96.
[37] N. Parolini, Computational Fluid Dynamics for Naval Engineering Problems, PhD Thesis, Number 3138, Ecole Polytechnique Federale de Lausanne (EPFL), 2004.
[38] N. Parolini and E. Burman, A finite element level set method for viscous free-surface flows, Applied and Industrial Mathematics in Italy, Proceedings of SIMAI 2004, 417–427, World Scientific, 2005.
[39] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan and L. Tobiska, Quantitative benchmark computations of two-dimensional bubble-dynamics, MOX-Report No. 23/2008, 2008.