PH.D. RESEARCH PROPOSAL:

Development of a Incompressible Multiphase Navier Stokes Solver for Non-Orthogonal, Curvilinear Adaptive Mesh System

Somdeb Bandopadhyay

August 03, 2016

Supervisor:
Yi Ju CHOU

Institute of Applied Mechanics

National Taiwan University

Contents

List of Figures

1 Literature Review & Motivation

Numerical simulations of multi-scale, multi-phase flows require a efficient model to capture/track the interface and sufficiently small grid cells to resolve the finest length scale. Examples of such cases include liquid jet atomization and droplet breakup. As a consequence of rapid flow evolution, the detailed fluid physics and dynamics can only be resolved by cells several orders of magnitude smaller than the length scale of the fluid at its initial state. Thus the numerical model used for the interface advection must be efficient to capture these multi-scale dynamics. In practice adaptive mesh refinement (AMR) algorithm, alogwith a efficient interface advection model is used to efficiently resolve the detailed flow physics in regions with interfacial boundaries.

In this proposed research, we intend to develop a multiphase navier stokes with adaptive mesh facility. The structure of this proposal is described here. At first, the state of art of relevant numerical methods are discussed. Henceforth, we try to explain the motivation and importance of this proposed work. In chapter 2 we provide the research Plan in brief. This is only the first part of this report. If this research proposal is approved, a much more detailed description will be reported in due time.

1.1 Literature Review: Numerical Approach for Multiphase Flow

The numerical formulation of the multiphase problems can be performed by Lagrangian method, Eulerian method or combination of both. The Lagrangian formulations lead to the use of moving, boundary conforming meshes for the interface. Considering the expected deformation of the interface, such approach would be unwise to pursue. The Eulerian approach, on the contrary, can be applied on a fixed mesh framework. In following sections, we shall briefly review the evolution and development of different numerical methods for multiphase flow problem. Detailed formulation of every method is beyond the scope of this review and only a generalized descriptions are provided here.

1.1.1 Early Works : The MAC Method

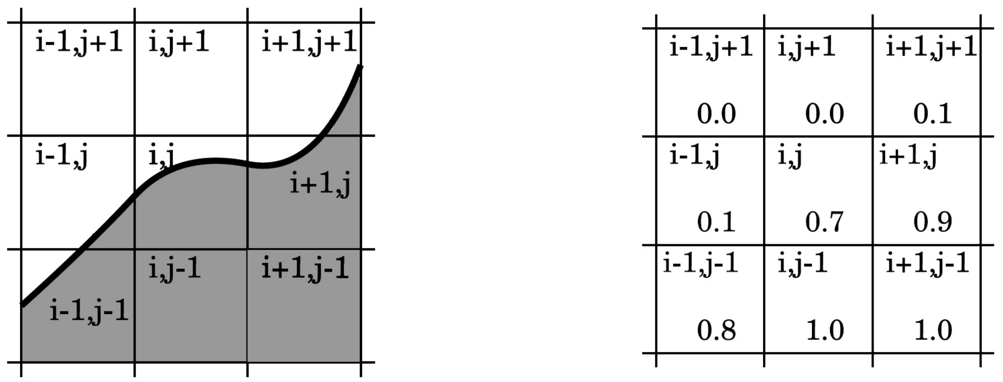
The oldest approach to numerically solve multiphase navier-stokes equation is the Marker-and-Cell (MAC) method developed at Los Alamos by Harlow and his collaborators to solve the free surface flow problems. In this method the fluid is identified by marker particles distributed throughout the fluid region and the governing equations are solved on a regular grid which covers both the fluid-filled and the empty part of the domain. Notably, although the prime objective to develop the MAC method was free surface flow problem, it was also the first method to successfully solve the Navier-Stokes equation using the primitive variables. The staggered grid used in this approach was a novelty at that time. The method was introduced in Harlow and Welch (1965) and two sample computations of the classical dam-breaking problem were shown in that first paper. Afterwards it was used by Harlow and Welch (1966) to examine the Rayleigh-Taylor problem and Harlow and Shannon (1967) to study the splashing phenomena. In the original implementation, the MAC method assumed a free surface and as such, only one fluid was under consideration. However, the Los Alamos group soon realized that the method can be applied to two-fluid problems. This revelation lead to the published work of Daly (1969b) where the author computed the evolution of the Rayleigh-Taylor instability for finite density ratios. Afterwards, surface tension was added by Daly (1969a) and the method was again used to examine the Rayleigh-Taylor instability. The MAC method quickly attracted a number of research groups who used it to study several problems including free surface waves [Chan and Street (1970)], the oscillations of an axisymmetric droplet [Foote (1973)], the collision of a droplet with a rigid wall [Foote (1975)], and the collapse of a cavitation bubble [Chapman and Plesset (1972); Mitchell and Hammitt (1973)].

1.1.2 VOF Method

The next generation of methods for multiphase flows evolved gradually from the MAC method. It was noted in the work by Harlow and Welch (1965) that the marker particles could cause inaccuracies. As a result, the particles were replaced by a marker function and the Volume-of-Fluid (VOF) method was born. VOF was first discussed in the published work of Hirt and Nichols (1981) but the approach was already used in previous works [De Bar (1974); Noh and Woodward (1976)]. A cell-averaged marker function was used To resolve the issue of numerical diffusion while advecting a sharp marker function. Moreover, the interface is "reconstructed" in the VOF method in such a way that the marker does not start to flow into a new cell until the current cell is full. In the early implementation of VOF, the interface in each cell was simply assumed to be a vertical plane for advection in the horizontal direction and a horizontal plane for advection in the vertical direction. This relatively crude reconstruction often lead to large amount of "floatsam and jetsam" (small unphysical droplets that break away from the interface) that degraded the accuracy of the computation. To improve the representation, Youngs (1982), Ashgriz and Poo (1991) and others introduced more complex reconstructions of the interface, representing it with a line (two dimensions) or a plane (three dimensions) that could be oriented arbitrarily in such a way as to best fit the interface. This increased the complexity of the method considerably but resulted in greatly improved advection of the marker function. However, even with higher-order representation of the fluid interface in each cell, the accurate computation of surface tension remained a major problem. This problem was countered by Brackbill et al. (1992) who showed that the curvature (and hence surface tension) could be computed by taking the discrete divergence of the marker function. A "conservative" version of this "continuum surface force" method was developed by Lafaurie et al. (1994). The VOF method has been improved in various ways by a number of authors. Apart from better reconstruction scheme for the interface [Rider and Kothe (1998); Scardovelli and Zaleski (2000); Aulisa et al. (2007)] and computation of the surface tension [Renardy and Renardy (2002); Popinet (2009)], more advanced advection schemes for the momentum equation and better solvers for the pressure equation have been introduced [e.g Rudman (1997)]. Other improvements of VOF method includes the use of sub-cells to keep the interface as sharp as possible [Chen et al. (1997)].

Being one of the oldest methods, VOF has been thoroughly investigated over past years to seek possible improvements. Among the most comprehensive reports in this context, the works by Rudman (1997) Rudman and by Gopala and van Wachem (2008). In both papers the ability to keep the interface sharp and the mass conserved has been studied with simplified advection and shear flow cases and a case capturing the progression of the Rayleigh-Taylor instability. Rudman (1997) reported the superior behavior of the direction split method proposed by Youngs (1982) compared to the Simplified Line Interface Calculation (SLIC) method [Noh and Woodward (1976)], the original VOF method [Hirt and Nichols (1981)] and the flux-corrected transport (FCT) method proposed by Rudman himself. Gopala and van Wachem (2008) considered the Lagrangian Piecewise Linear Interface Construction (PLIC) introduced in Van Wachem and Schouten (2002), the CICSAM [Ubbink (1997)] and the inter-gamma differencing scheme [Jasak and Weller (1995)] to be preferable over the above mentioned FCT method. Xiao et al. (2005) have presented a novel multi-dimensional algebraic VOF method where they used the hyperbolic tangent function to compute the numerical flux for the fluid fraction function. Based on the numerical approach, this version of VOF is named as WLIC (Weighted Linear Interface Calculation). Yokoi (2007) has presented a simplified version of THINC vased WLIC-VOF. The WLIC-VOF has been shown to have same order of accuracy as PLIC-VOF Xiao et al. (2011) although being relatively simpler to implement. Recently Aniszewski et al. (2014) has presented a comprehensive discussion between PLIC and WLIC methods.

Another major direction in recent research on VOF methods has been focused on the interface reconstruction methods. Recently Dyadechko and Shashkov (2005) has presented a higher order formulation of volume tracking approach similar to VOF. This method, named as MOF (Moment Of Fluid) is further analyzed by Ahn and Shashkov (2009) and others to be applied for general multiphase flow. Some other notable works in reconstruction methods are Aulisa et al. (2007), Diot and François (2016) and Kawano (2016).



(a) Characteristic function

(b) Volume fraction / Color function

Figure 1: Characteristic Function and Volume Fraction

A brief introduction is given here regarding the numerical formulations used in VOF. Let's consider two phase flow in two-dimensions with fluid A and fluid B. We introduce a characteristic function $\chi(x, y)$ and a volume fraction or color function C_{ij}, defined as

$$\chi(x,y) = \begin{cases} 1, & \text{for the fluid A at the point } (x,y) \\ 0, & \text{for the fluid B at the point } (x,y) \end{cases} \tag{1.1}$$

$$C_{ij} = \frac{1}{\Delta x \Delta y} \int_0^{\Delta x} \int_0^{\Delta y} \chi(x,y) dxdy \tag{1.2}$$

Note that $\chi(x, y)$ is the value defined at a point (x, y) and is not a value on the computational grid. On the contrary the color function C_{ij} is defined for each grid cell. Being a continuous function, the value of χ is value 0 or 1, however, the value of C_{ij} is defined as $0 \leq C_{ij} \leq 1$. As we can see, the interface is defined at $\Gamma(x, y)$ where $\chi(x, y)$ changes it's value from 0 to 1 or vice-versa. Also, this $\chi(x, y)$ will be advected as a passive scalar in the flow field i.e.

$$\left. \begin{array}{c} \dfrac{D\chi}{Dt} = 0 \\ \dfrac{\partial \chi}{\partial t} + \dfrac{\partial \chi u}{\partial x} + \dfrac{\partial \chi v}{\partial y} = 0 \end{array} \right\} \tag{1.3}$$

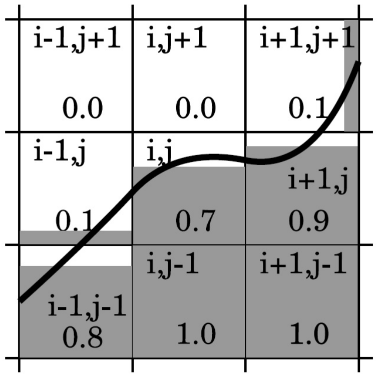
χu and χv represents fluxes of χ through the boundaries of control volume. Scardovelli and Zaleski (2000) applied a split approach towards VOF advection. Their formulation calculates fluxes separately along axes and thus, is ideal for parallel implementation. Combining equations from (1.1), (1.2) and (1.3) the split approach gives us:

$$\frac{C_{ij}^{n+1}-C_{ij}^{n}}{\Delta t}=\frac{1}{\Delta x}\left[(F_x)^n_{i-\frac{1}{2},j}-(F_x)^n_{i+\frac{1}{2},j}\right]+\frac{1}{\Delta y}\left[\left(F_y\right)^n_{i,j-\frac{1}{2}}-\left(F_y\right)^n_{i,j+\frac{1}{2}}\right] \tag{1.4}$$

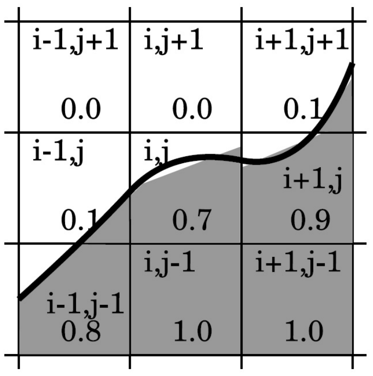
The terms F_x and F_y represents the flux of χ in x-direction and y-direction along the control volume boundaries. Equation (1.4) essentially represents the discretized coupling between $\chi(x,y)$ and C_{ij} in a 2D computational framework.

If values of χ_{ij} is known, one can easily use equation (1.2) to calculate C_{ij} . However, we also need to estimate $\chi(x,y)$ to compute the curvature of the interface which is used to compute the surface tension terms. The problem in the VOF method is attributed to reconstruction of $\chi(x,y)$ from the volume fraction C_{ij} . As described before, two typical methods to reconstruct $\chi(x,y)$ are the SLIC (simple line interface calculation)[Noh and Woodward (1976)] and the PLIC (piecewise linear interface calculation) method based on works of Youngs (1982). In the SLIC method $\chi(x,y)$ is reconstructed by a line along x or y coordinate as shown in Fig. 2a. An advantage of the SLIC method is the easement of implement. In the PLIC method, χ_i, j is calculated based on an diagonal line/plane as shown in Fig. 2b. The WLIC method as presented in Xiao et al. (2005) and Yokoi (2007) is shown in Fig. 3.

In all versions of VOF methods, a modified form of the Navier-Stokes equation is used for the entire domain:



(a) SLIC Reconstruction



(b) PLIC Reconstruction

Figure 2: SLIC and PLIC

$$\frac{\partial \rho \vec{u}}{\partial t} + \nabla \cdot (\rho \vec{u}\vec{u}) = -\nabla \cdot p + \nabla \cdot (2\mu \vec{E}) + \vec{f}_b + \vec{f}_s \tag{1.5}$$

where the density and viscosity fields are defined in discrete form as :

$$\left.\begin{aligned} \rho_{ij} &= \rho_A C_{ij} + \rho_B (1 - C_{ij}) \\ \mu_{ij} &= \mu_A C_{ij} + \mu_B (1 - C_{ij}) \end{aligned}\right\} \tag{1.6}$$

$\vec{u}$, $\vec{f}_b$ and $\vec{E}$ represent the velocity field, body force and the stress tensor respectively. The term $\vec{f}_s$ describes the Laplace pressure acting at the surface of discontinuity and is given by:

$$\vec{f}_s = \sigma \kappa \vec{n} \delta_\Gamma \tag{1.7}$$

where the interface is indicated by the Dirac delta function δ_Γ. For numerical implementation, the term $\vec{f}_s$ is replaced by a continuum surface force (CSF):

$$\vec{f}_v = \sigma \kappa \nabla \chi \tag{1.8}$$

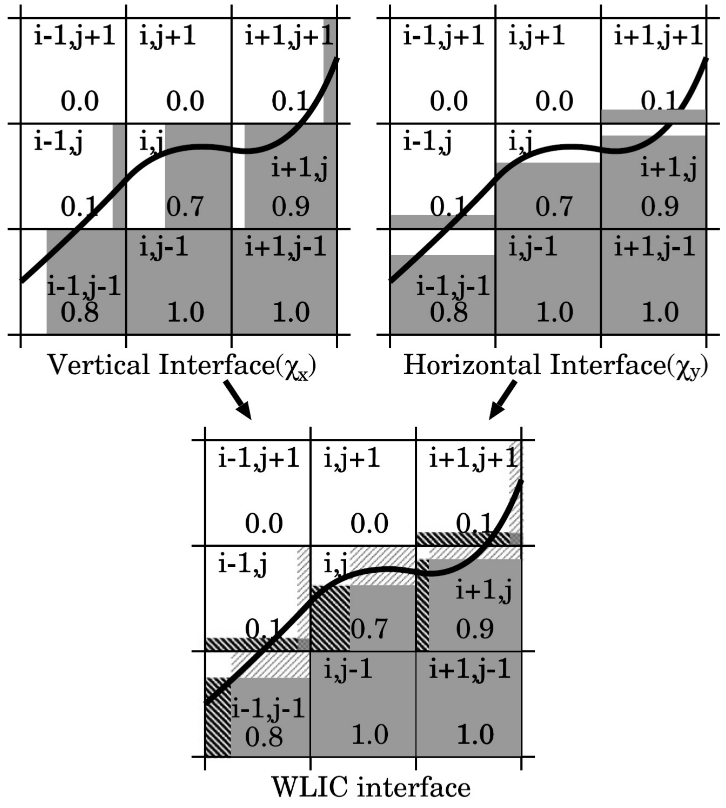


Figure 3: WLIC Reconstruction

$\vec{f}_v$ tends to $\vec{f}_s$ as the thickness of the interface region tends to zero. The curvature κ is calculated from the approximated characteristic function. This can't be done directly as it leads to large spatial oscillations. Recursive smoothing is employed to get a sufficiently smooth characteristic function, $\hat{\chi}$. The curvature can then be calculated from the smoothed function using:

$$\kappa = \nabla \cdot \vec{n} = \nabla \cdot \left(\frac{\nabla \hat{\chi}}{|\nabla \hat{\chi}|} \right) \tag{1.9}$$

At the solid boundaries, velocity component normal to the solid wall is zero. At the triple-contact line, Young's law determines the contact angle:

$$cos\theta = \frac{\sigma_{B,s} - \sigma_{A,s}}{\sigma} \tag{1.10}$$

where $\sigma_{B,s}$ is the fluid B/solid interfacial tension, $\sigma_{A,s}$ is the fluid A/solid interfacial tension. For imposing the contact angle in the simulation, this is equivalent to imposing the boundary condition:

$$\vec{n}_{\Gamma_s} = \vec{n}_s cos\theta + \vec{t}_s sin\theta \tag{1.11}$$

$\vec{t}_s$ is the unit tangential vector pointing into the fluid A part of the domain.

1.1.3 Front Tracking Method

The theoretical essence behind the MAC and the VOF methods directly or indirectly influenced many new approaches in the early nineties. Unverdi and Tryggvason (1992) introduced a Front-Tracking method for multiphase flows where the interface was marked by connected massless marker points. The markers are used to advect the material properties (such as density and viscosity) and to compute surface tension, but the rest of computations is done on a fixed grid as in the VOF method. In front tracking methods the interface is defined explicitly, usually by marker particles distributed on interface. The interface is then evolved by solving the ordinary differential equation for each particle:

$$\frac{\partial X}{\partial t} = \vec{u}(X) \qquad \text{for X} \in \Gamma \tag{1.12}$$

In order to calculate normals and curvatures one needs to keep track of which particle is neighbor to which particle. Particles might move too close together or too far apart, resulting in numerical difficulties. To avoid this, the particles might have to be redistributed along the interface. Since the particles move independently of each other, oscillations in the interface might occur. Some regularizing technique has to be applied to suppress such oscillations. Interpolations also have to be done to interpolate the interface to the fixed grid used for the velocity and pressure. Special care also has to be taken when merging or breakup of interfaces occur. Other examples for the use of marker points include the Immersed-Boundary Method of Peskin (1977) for one-dimensional elastic fibers in homogeneous viscous fluids and the Vortex-in-Cell method of Tryggvason and Aref (1983) for two-fluid interfaces in a Hele-Shaw cell. Tryggvason and collaborators have used the front tracking scheme to explore a large number of problems.

1.1.4 Level Set Method

Other numerical methods for multiphase flow includes Level-Set, CIP, and the Phase-Field methods to track fluid interfaces on fixed grids. The Level-Set method was pioneered by Osher and Sethian (1988), but its first use to track fluid interfaces was performed in the the work of Sussman (1994) and Chang et al. (1996) who used level set to simulate the rise of bubbles and the fall of droplets in two-dimensions. An axisymmetric version was applied by Sussman and Smereka

(1997) to examine the behavior of bubbles and droplets. Unlike the VOF method, where a discontinuous marker function is advected with the flow, in the Level-Set method a continuous level-set function is used. The interface is then identified with the zero contour of the level-set function. To reconstruct the material properties of the flow a marker/color function is constructed from the level-set function. The marker function is given a smooth transition zone from one fluid to the next, thus increasing the regularity of the interface over the VOF method where the interface is confined to only one grid space. However, this mapping from the level-set function to the marker function requires the level-set function to maintain the same shape near the interface.To deal with this problem, Sussman (1994) introduced a reinitialization procedure where the level-set function is adjusted accordingly, such that, its value is equal to the shortest distance to the interface at all times. This step was essential to evolve Level-set m/d applicable for fluid-dynamics simulations. Computation of surface tension is performed in the same way as in the continuous surface force technique introduced for VOF methods by Brackbill et al. (1992). The early implementation of the Level-Set method did not conserve mass very well and a number of improvements has been suggested till date. Sussman et al. (1998) and Sussman and Fatemi (1999) modified the level set formulation to improve mass conservation, Sussman et al. (1999) applied level-set with adaptive grid refinement and a hybrid VOF/Level-Set method was developed by Sussman and Puckett (2000).

As described above, in level set method the interface is implicitly defined by a marker function ϕ. The choice of ϕ is made such that, it changes sign at the interface. Thus, it has the following properties:

$$\phi(x,y) = \begin{cases} > 0, & \text{when point } (x,y) \text{ filled with fluid A} \\ = 0, & \text{when point (x,y) is at the interface} \\ < 0, & \text{when point } (x,y) \text{ filled with fluid B} \end{cases} \tag{1.13}$$

Since the tracked interface is the zero level set of the function ϕ , the interface can be tracked by solving the following equation:

$$\frac{\partial\phi}{\partial t} + \vec{u}\cdot\nabla\phi = 0 \tag{1.14}$$

Actually only the normal component of $\vec{u}$ defined by $u_N = \vec{u}\cdot\frac{\nabla\phi}{|\nabla\phi|}$ will affect the movement of the interface Γ. Thus a modified form of equation (1.14) can be derived for this purpose:

$$\frac{\partial\phi}{\partial t} + u_N\cdot\nabla\phi = 0 \tag{1.15}$$

ϕ will , as mentioned before, not remain a signed distance function as time evolves. A reinitialization is therefore needed to make sure that ϕ remains a distance function. This reinitialization can also be formulated as a PDE. Given an initial ϕ_0 with the correct position of the zero contour

$$\frac{\partial\phi}{\partial t} + \left[sgn(\phi_0)\right](|\nabla\phi| - 1) = 0 \tag{1.16}$$

can be solved to steady state. This was used by Sussman (1994) where the level set method was first used for incompressible two phase flow calculations. In contrast to the characteristic function used in the volume of fluid method, ϕ is smooth across the interface Γ and because of this, standard higher order techniques (e.g ENO, WENO) can be used to solve equation (1.14).

Similar to VOF methods, a regularization of fluid properties can be done. We now define a Heaviside function $H()$, given by:

$$H(\phi) = \begin{cases} 0, & \phi < 0 \\ \frac{1}{2} + \frac{\phi}{2\epsilon} + \frac{1}{2\pi} sin\left(\frac{\pi\phi}{\epsilon}\right), & -\epsilon \le \phi \le \epsilon \\ 1, & \phi > \epsilon \end{cases} \tag{1.17}$$

The value of ϵ indicates a finite thickness for the interface. Similar to equation (1.6) we can get

$$\left.\begin{aligned} \rho &= \rho_A H(\phi) + \rho_B (1 - H(\phi)) \\ \mu &= \mu_A H(\phi) + \mu_B (1 - H(\phi)) \end{aligned}\right\} \tag{1.18}$$

and apply it to the modified navier stokes equation.

The curvature calculation of the interface can be performed based on ϕ similar to the previous calculations in VOF. κ=κ_ϕ is given by:

$$\kappa_\phi = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \tag{1.19}$$

For 3D, this becomes:

$$\kappa_\phi = \frac{1}{|\nabla\phi|^3}\Big(\phi_x^2\phi_{yy} - 2\phi_x\phi_y\phi_{xy} + \phi_y^2\phi_{xx} + \phi_x^2\phi_{zz} - \\ 2\phi_x\phi_z\phi_{xz} + \phi_z^2\phi_{xx} + \phi_y^2\phi_{zz} - 2\phi_x\phi_y\phi_{xy}\Big) \tag{1.20}$$

Recently, Enright et al. (2002) presented a modified version of level set method, where markers were used to correct the directed level set function. In this approach, named as MLS (Marker Level Set) or Particle Level Set (PLS), massless marker particles are utilized to correct the level set function in the under-resolved areas of the interface in order to preserve mass. Two sets (positive and negative) of particles are randomly placed within a band across the interface. Fig. 4 gives a presentation of the approach. It is like a rope-pulley system where the radius of pulley (particles) is defined by :

$$r_p = \begin{cases} r_{max}, & \text{if } S_p\phi(x_p) > r_{max} \\ S_p\phi(x_p), & \text{if } r_{min} \le S_p(x_p) \le r_{max} \\ r_{min}, & \text{if } S_p(x_p) < r_{min} \end{cases} \tag{1.21}$$

Here, value of the signed function S_p is +1 or -1 depending of which side of interface the particle is located. After temporal evolution, a check is performed to identify whether any particle has switch side with respect to interface. For each escaped particle, a new level set function is computed as $\phi_p = S_p(r_p - |x - x_p|)$. Afterwards, ϕ^+ and $\phi^-$are calculated as:

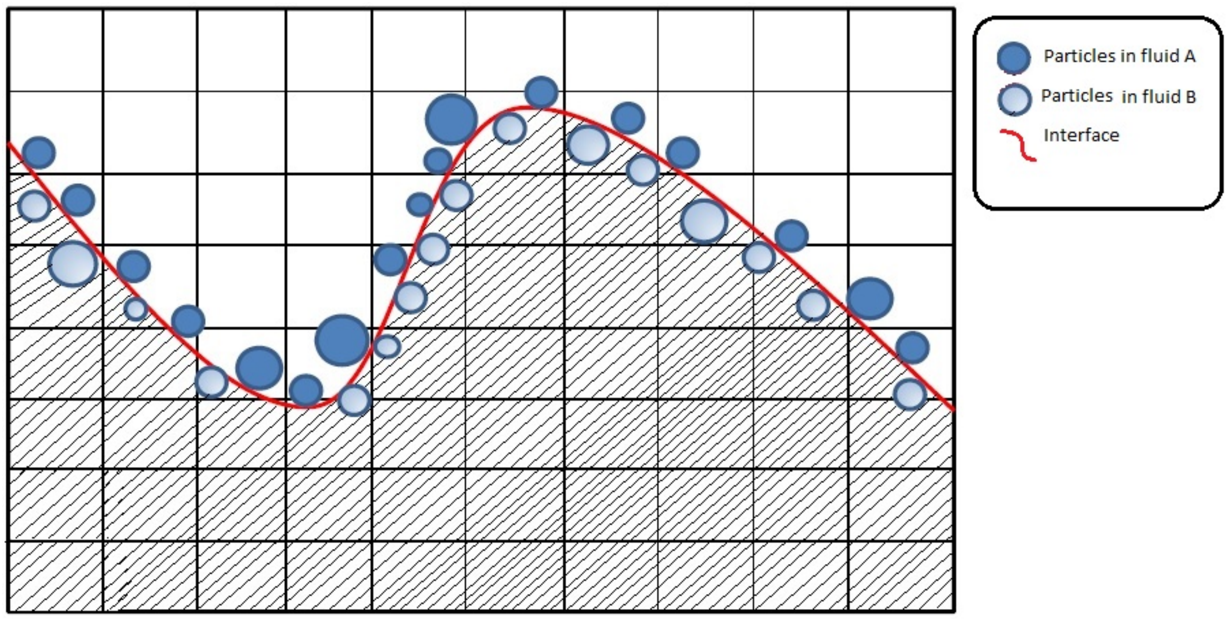


Figure 4: Marker Level Set/ Particle Level set

$$\left.\begin{aligned} \phi^+ &= \max(\phi_p, \phi_A) \qquad \text{for fluid A side} \\ \phi^+ &= \max(\phi_p, \phi_B) \qquad \text{for fluid B side} \end{aligned}\right\} \tag{1.22}$$

Finally the final correction is performed as:

$$\phi = \begin{cases} \phi^+, & \text{if } |\phi^+| \le |\phi^-| \\ r_{min}, & \text{if } |\phi^+| > |\phi^-| \end{cases} \tag{1.23}$$

The correction step in MLS is further discussed and improved by Gaudlitz and Adams (2008), Wang et al. (2009) and others.

1.1.5 CIP and Phase Field Method

The Constrained Interpolated Propagation (CIP) method was introduced by Takewaki et al. (1985). In the CIP method the transition from one fluid to another is formulated by a cubic polynomial. Both the marker function and its derivative are then updated to advect the interface. In addition to the simulation of two-fluid problems, this method has been used for a number of more complex applications, such as those involving floating solids by Yabe et al. (2001).

The Phase-Field approach was introduced by [Kobayashi (1992, 1993)] to model solidification. In the Phase-Field method the governing equations are modified to describe the dynamics of the smoothed region between the different fluids in a thermodynamically consistent way. Computationally, the overall effect of the modification is to provide an "anti-diffusive" term that keeps the interface reasonably sharp. Although there are considerable similarities between Phase-Field and Level-Set methods, the fundamental ideas behind these two methods are very different. In the Level Set method the smoothness of the phase boundary is artificial and introduced for numerical reasons only. On the contrary, in Phase-Field methods the transition zone is real, although it is made much thicker than it should be for numerical reasons. So in phase field model, we apply proper thermodynamic conditions in an artificially thick interface. For the motion of two immiscible fluids with a sharp interface, this condition leads to additional resolution requirement that is more stringent than for other "one-fluid" methods. The Phase-Field methods are advantageous for problems

1. Interface Capturing Methods [Quirk (1994)]
2. Interface Tracking Methods
    - A. Front Tracking Methods [Unverdi and Tryggvason (1992)]
    - B. Volume Tracking Methods
        - I. Marker and Cell (MAC) [Hirt and Nichols (1981)]
        - II. Volume Of Fluid (VOF)
            - a) Simple Line Interface Calculation (SLIC) [Noh and Woodward (1976)]
            - b) Piecewise Linear Interface Calculation (PLIC) [Youngs (1982)]
            - c) Weighted Line Interface Calculation (WLIC) [Xiao et al. (2005)]
            - d) Flux Corrected Transport (FCT) [Rudman (1997)]
            - e) Constrained Interpolation Profile (CIP) [Takewaki et al. (1985)]
        - III. Moment Of Fluid (MOF) [Dyadechko and Shashkov (2005)]
        - IV. Level Set (LS) [Osher and Sethian (1988)]
        - V. Hybrid Methods
            - a) Coupled Level-Set Volume-Of-Fluid (CLSVOF) [Sussman and Puckett (2000)]
            - b) Marker Level-Set (MLS) [Enright et al. (2002)]
    - C. Phase-Field Methods [Kobayashi (1992)]

Figure 5: Classification of numerical methods for multiphase flow

where small-scale physics are of high importance, since it is difficult to do so in the sharp interface limit. Compared to VOF, Front-Tracking m/d or Level Set method, use of Phase-Field approach for fluid dynamic simulations is relatively limited [Jacqmin (1999); Jamet et al. (2001)].

1.1.6 Sharp-Interface Methods

In all the "one-fluid" methods described above, the accuracy is generally compromised due to the numerical thickness of the interface. Therefore several attempts has been made to treat the interface as "fully sharp". Glimm and his collaborators [Glimm et al. (2001)] applied local modification to the grids near the interface in such a way that the interface coincided with a cell boundary. Another example is the "cut-cell" methods which is used to include complex bodies in simulations of inviscid flows by Quirk (1994) and Powell (1998).

In the "ghost fluid" method introduced by Fedkiw et al. (1999) the interface is marked by advecting a level-set function. Fictitious values are assigned to grid points near the interface for the numerical approximations of derivatives (i.e where the finite difference stencil involves values from both sides of the interface). These fictitious values are obtained by extrapolation and a there exists different approaches to do so.

Udaykumar et al. (2001) represented complex solid boundaries on a regular grid by merging cells near the interface and using polynomial fitting to find field values at the interface. This method has been extended and well documented in Marella et al. (2005), Liu et al. (2005) and Yang and Udaykumar (2005).

The "Immersed Interface" method of Lee and Leveque (2003) modifies the numerical approximations near the interface by explicitly incorporating jump condition across the interface into finite difference equations. While this is easily done for relatively simple jump conditions, the simplicity is compromised for complex situations. As a result Lee and Leveque (2003) limited their implementation for fluids with the same viscosity.

A classification of the reviewed methods is presented in Fig (5).

1.2 Literature Review: Adaptive Mesh Refinement

Recent evolutions in computer technology have made it easier to perform numerical simulations using CFD, especially on highly parallel computing machines. Along with the advancements in computer technology, the numerical methods for the solution of fluid flow have also evolved. The advanced AMR (Adaptive Mesh Refinement) strategies and the load balancing techniques associated with them are very common in nowadays simulations. The AMR technique is a logical extension of the concept behind invoking nonuniform grids to efficiently utilize computer resources. Grids of higher resolution are used where more accurate solution is desired. There are many scientific and engineering applications enriched with multiscale physics phenomena for which AMR routines are most suited. For instance, in complex fluids, higher resolution is preferred near the interface between different phases because of the rapid changes in the fluid properties. As noted previously, all conventional numerical approaches for multiphase flow, suffer from accuracy issues due to the numerical thickness of interface. AMR method counters these inaccuracies by dynamically refining the mesh near the interface. There exists a large variety of approaches to introduce mesh adaptivity which differ in many aspects such as mesh topology, refinement criteria and data structure management. A complete review is beyond the scope of this proposal. We shall only focus on structured adaptive mesh refinement procedures in this proposal. Constructing a parallel AMR algorithm is a demanding and complicated subject. There is an ongoing research on paralleling AMR algorithms and the load balancing techniques associated with them.

1.2.1 Structured Adaptive Mesh Refinement

There are two main approaches that have been developed for Adative Mesh refinement in structured grid, namely Block Structured Adative Mesh Refinement, Quadtree/Octree Based Adaptive Mesh Refinement and Cell-based Structured Adaptive Mesh Refinement .

The first approach utilizes a block-structured style of mesh refinement [e.g. Berger and Oliger (1984)], in which a sequence of nested structured grids at different hierarchies or levels (cell sizes are the same for all grids at the same hierarchy or level) are overlapped with or patched onto each other. A tree-like (or, directed acyclic graph) data structure is usually used to facilitate the communication (transfer of information) between the regular structured grids at the various hierarchies. It should be noted that each node in this tree-like data structure represents an entire grid rather than simply a cell and, as a result, efficient solvers for structured grids can be used to solve the system of algebraic equations resulting from the discretization of the governing equations on each node. This strategy, however, is not flexible in adaptivity nor is it optimal in terms of computational cost and memory management because overlapping rectangular patches may cover parts of the computational domain that does not need to be resolved resulting in a waste of computer memory and time. Moreover, the entire grid hierarchy must be rebuilt once refinement/coarsening is carried out.

In the second approach for AMR, the hierarchy of grids at different levels is represented by a quad-tree/oct-tree data structure in two/three dimensions (2-D/3-D), with each node representing an individual grid cell (rather than a block of grid cells as in the first approach). Because the mesh is only locally refined in the region where it is required, the second approach results in improved computational efficiency, increased storage savings, and better control of the grid resolution in comparison with the first approach. In a conventional oct-tree discretization and representation, such as that used by MacNeice et al. (2000), the connectivity information between an individual cell and its neighbors needs to be stored explicitly, with each cell requiring 19 words of computer memory to maintain the oct-tree data structure. In addition to the large memory overhead required to maintain this oct-tree data structure, it is also very difficult to parallelize owing to the fact that the neighbors of each cell are connected (linked) each other. Consequently, removing or adding one cell in the process of adaptive mesh refinement will affect all the neighboring cells of this one cell. If the pointers to the neighboring cells are not stored explicitly, the oct-tree must be traversed to find these neighboring cells. On average, it is expected that two levels of the oct-tree must be traversed before a neighbor can be found, but in the worst-case scenario it may be necessary to traverse the entire tree to access the nearest neighbors. Tree traversals of this type can extend from one processor to another, making it difficult to vectorize and parallelize these operations. Initial approach for Oct/Quadtree based AMR was performed by DeZeeuw and Powell (1993). Misleadingly, they argued that finding the neighbors of a given cell was better than storing eight integer words per cell. Therefore, they preferred to do a tree search to find the neighboring cells rather than storing the cell-neighbor information via pointers, which cost them about 25% of their computation time. This issue was countered by the introduction of fully threaded tree (FTT) [e.g Khokhlov (1998)]. FTT data structure was developed to reduce the memory overhead required to maintain the information embodied in a tree structure and to facilitate rapid and easy access to the information stored in the tree. In the FTT structure, all cells are organized in groups referred to as octs. Each oct maintains pointers to eight (child) cells and to a parent cell. Each oct stores information about its level (equal to the level of the parent cell) and maintains the information about its position in the computational domain. Furthermore, each oct maintains pointers to the parent cells of the six neighboring octs, and each cell of the oct maintains a pointer to its child oct (or to a null pointer if the cell has no children). The memory requirements for the FTT structure is 19 words per oct or $2\frac{3}{8}$ words per cell. The memory overhead is significantly reduced compared to that used in the conventional oct-tree structure. Another advantage of the FTT structure is that the actual number of traversed levels required to find a neighbor never exceeds one. However, the FTT is still based on a tree data structure. In a tree-based code for AMR, a complete tree traversal starting from the coarsest cell (i.e., root cell) is frequently required to access a given cell in the adaptive mesh. To avoid a complete traversal of the tree from the root cell, a linked list is frequently used to improve the access efficiency in the tree. However, the use of a linked list incurs a computer memory overhead. Furthermore, it may be very difficult to access an arbitrary cell in the tree using a linked list in a parallel code, owing to the fact that the parent of a particular cell may not even reside on the same processor.

Ji et al. (2010) proposed the cell-based structured AMR method which replaces the tree structure with a hash table. Accordingly, the computer memory required for storing the data structure is reduced. The penalty paid for this saving in memory is the computation overhead due to calculation of the connectivity information such as finding neighbors of a particular cell. Needless to say, the hashing technique relies on external resources for maintaining the data structure and connectivity information, which prevents the user from having a transparent and stand-alone AMR routine.

1.2.2 Available AMR Libraries

SAMR codes all rely on the same fundamental concept, viz. that the solution can be computed in different regions of the do- main with different spatial resolutions, where each region at a particular resolution has a logically rectangular structure. In some SAMR codes the data is organized by level, so that the description of the hierarchy is fundamentally defined by the union of blocks at each level; while others organize their data with unique parentâĂŞchild relationships. Along with the spatial decomposition, different codes solving time-dependent equations make different assumptions about the time stepping, i.e., whether blocks at all levels advance at the same time step, or blocks advance with a time step unique to their level. Finally, even when frameworks are used to solve exactly the same equations with exactly the same algorithm, the performance can vary due to the fact that different frameworks are written in different languages, with different choices of data layout, and also differ in other implementation details. However, despite their differences in infrastructure and target domain applications, the codes have many aspects that are similar, and many of these codes follow a set of very similar software engineering practices.

Here we review some available AMR libraries. The major problem with most of these codes are the

Infrastructure	Description
Chombo	Consisted of four core modules: BoxTools, AMRTools, AMRTimeDependent, AMRElliptic. The load balance strategy follows Kernighan-Lin multilevel partitioning algorithm
ParaMesh	Extends serial code to parallel code based on partitioning octree representation of adaptive grid structure with predefined blocksizes
SAMRAI	Object oriented framework (based on LPARX) with patches mapped across processors at each level.
BoxLib	Combination of C++/Fortran90 subroutines, contains support for both explicit and implicit grid-based operations as well as particles on hierarchical adaptive meshes. Single-level and multi-level multigrid solvers are included for cell-based and node-based data.

1.2.3 Application of AMR in Multiphase Flow

Popinet (2003) and Fuster et al. (2009) has applied the octree type adaptive mesh framework for VOF method. Recently Tan (2016) used a modified version of the same numerical framework to analyze free-surfaces flow problems related to thermal inkjet technology. Laurmaa et al. (2016) has developed a semi-Lagrangian VOF approach in the adaptive mesh framework and used it to study free surface flow. As mentioned previously Sussman et al. (1999) applied AMR for level-set method. Recent works in this context include Losasso et al. (2006) and Kolomenskiy et al. (2016). Application of adaptive mesh for front tracking method and phase field method can be found in Pivello et al. (2014) and Ceniceros et al. (2010) respectively.

1.3 Motivation and Novelty of Current Work

The main objective of current research is to develop a multiphase flow solver with AMR facility for nonorthogonal, dynamically adaptive, curvilinear mesh. Although the current work is focused on multiphase flow, the preparation of AMR algorithm will have a generalized impact in overall incompressible flow analysis.

The main motivation of this work, is to have a high-order accurate multiphase flow solver which can be applied to perform Direct Numerical Simulation in complex flow domain. We note that, higher order methods for non-orthogonal, curvilinear mesh is not new. Particularly in aeroacoustics field such method has been studied thoroughly over last ten years. However, implementation of such methods in multiphase flow problems are yet to be addressed properly. The main focus in numerical multiphase flow research has been on orthogonal (e.g cartesian or cylindrical) and nearly-orthogonal grid system.Also immersed boundary methods are widely used for computation involving complex geometry of flow domain. However, IB methods inherently introduces error terms near the embedded boundaries and even if we consider the immense effort given to IB research, high resolution solution (DNS frequency) cannot be achieved with these methods. Till date, a fully non-orthogonal, high order framework for multiphase flow computation has not been developed as per our knowledge. Thus, this work would be first of it's kind.

Secondly, as already stated before, a multiphase flow solver/algorithm without AMR facility essentially is not applicable to analyze the multiscale dynamics. Although research in AMR has rich foundation , prime attention is given to cylindrical or cartesian mesh systems. In our work, the main focus will be on the implementation of AMR for non-orthogonal, curvilinear mesh system. The main challenge in this case is the nonlinear relation between the computational and physical grid. As such, the refinement and coarsening procedure needs to be revisited thoroughly to improve the conventional algorithms for these cases.

2 Research Plan

For the sake of consistency, this proposed research is divided into three parts:-

1. Preparation of Block Structured Adaptive Mesh Refinement framework for curvilinear grid system.
2. Development of the multiphase incompressible flow solver
3. Finally, development of accurate interface tracking method based on particle level set and it's implementation on the prepared solver

It should be noted that, development of SAMR framework for non-orthogonal curvilinear mesh is the most important part of this research. Hence at first, prime focus will be given to build this SAMR framework. C language is to be selected for this part, since it has better object-oriented programming facility. We intend to use a 3D advection equation with a wavepacket source will be used to see the efficiency of the AMR framework.

Once the computational framework is prepared, we shall use it to develop the incompressible flow solver. As we are yet to decide the scheme for interface tracking, description of the final part of this work will be provided later after thorough analysis and discussion with Prof. Chou.

In this chapter, we shall first discuss the foundation of non-orthogonal, curvilinear finite difference method which will be basis of the formulations. Afterwards, the proposed parallel SAMR framework will be introduced.

2.1 Finite Difference on Non-orthogonal curvilinear Mesh

Let (x,y,z) be the orthogonal cartesian co-ordinate system and (ξ,η,ζ) be the non-orthogonal curvilinear co-ordinates represented by the computational mesh.

Freestream preservation and metric cancellation must be addressed before we can apply high order finite difference schemes to nontrivial 3-D geometries. We first invoke the following identities:

$$I_1 = \left(\hat{\xi}_x\right)_\xi + \left(\hat{\eta}_x\right)_\eta + \left(\hat{\zeta}_x\right)_\zeta = 0 \tag{2.1a}$$

$$I_2 = \left(\hat{\xi}_y\right)_\xi + \left(\hat{\eta}_y\right)_\eta + \left(\hat{\zeta}_y\right)_\zeta = 0 \tag{2.1b}$$

$$I_3 = \left(\hat{\xi}_z\right)_\xi + \left(\hat{\eta}_z\right)_\eta + \left(\hat{\zeta}_z\right)_\zeta = 0 \tag{2.1c}$$

$$I_4 = \left(\frac{1}{J}\right)_\tau + \left(\hat{\xi}_t\right)_\xi + \left(\hat{\eta}_t\right)_\eta + \left(\hat{\zeta}_t\right)_\zeta = 0 \tag{2.1d}$$

The first three identities constitute a differential statement of surface conservation for a closed cell. The last metric identity (I_4) expresses volume conservation and is referred to in the literature as the geometric conservation law (GCL) [Thomas and Lombard (1979)]. In a finite-difference discretization, these identities must be satisfied numerically in order to ensure freestream preservation. For time-invariant coordinate transformations (i.e., fixed meshes), only the first three identities are applicable. Now, let's consider the conventional metric relations:

$$\left.\begin{aligned}\hat{\xi}_x &= y_\eta z_\zeta - y_\zeta z_\eta \\ \hat{\eta}_x &= y_\zeta z_\xi - y_\xi z_\zeta \\ \hat{\zeta}_x &= y_\xi z_\eta - y_\eta z_\xi\end{aligned}\right\} \tag{2.2}$$

These are associated with the x-components of surface area vectors. The corresponding metric identity for these relations is I_1 (Eqn. 2.1a), which must be satisfied numerically to ensure freestream preservation. Similar relations exist for the other two components of the surface area vectors. Evaluation of the y and z derivatives in Eqn.(2.2) using explicit or compact centered schemes does not satisfy the identity I_1 and therefore grid-induced errors appear in regions of large grid variation or near singularities. To address this problem for low-order schemes, Pulliam and Steger (1980) introduced a simple averaging procedure while Vinokur (1989) also proposed the use of finite-volume concepts. These approaches, which work very well for the second-order scheme, are not readily extendable to compact schemes, and in their present form are not suitable even for explicit higher-order formulations. An alternate and less used method for enforcing the metric identities was given by Thomas and Lombard (1979). Instead of introducing weighted averaging or invoking geometrical concepts, they rewrite the metric expressions in Eqn.(2.2), prior to discretization in the equivalent 'conservative' form :

$$\left.\begin{aligned}\hat{\xi}_x &= y_\eta z_\zeta - y_\zeta z_\eta \\ \hat{\eta}_x &= y_\zeta z_\xi - y_\xi z_\zeta \\ \hat{\zeta}_x &= y_\xi z_\eta - y_\eta z_\xi\end{aligned}\right\} \tag{2.3}$$

with similar relations for the remaining metric terms. This approach was proposed in the context of lower-order methods, but did not become popular because of the relatively simpler averaging procedure of Pulliam and Steger (1980). Although Eqn. (2.3) was not envisaged for use with higher-order or compact-difference-based methods, its 'conservative' differential form suggests its viability for the purpose and is therefore selected the proposed finite difference solver. A more thorough explanation can be found in the published work of Visbal and Gaitonde (2002).

2.2 Block Structured Adaptive Mesh with Quadtree

In the block-structured AMR, regions required for the AMR treatment in the simulation domain have a self-similar structure. We can explain this structure using a conventional Cartesian mesh. The self-similar block-structured domains for AMR are to be managed in a fully threaded tree (FTT) data structure which will allow recursive refinement on a block-by-block basis. Each block consists of a domain formed with the fixed number of cells with uniform cell size. A block in a different level of refinement in the FTT structure will have different cell configuration , keeping same number of cells. For instance, in one level higher, the cell size becomes half. In each block, we can incorporate the same uniform-cell simulation program of our interest and independently perform the simulation. Since each block has a common domain with the same number of cells because of self-similarity, what we need to consider is the cell size in each block depending on the refinement level which is given will be provided in the the FTT structure.

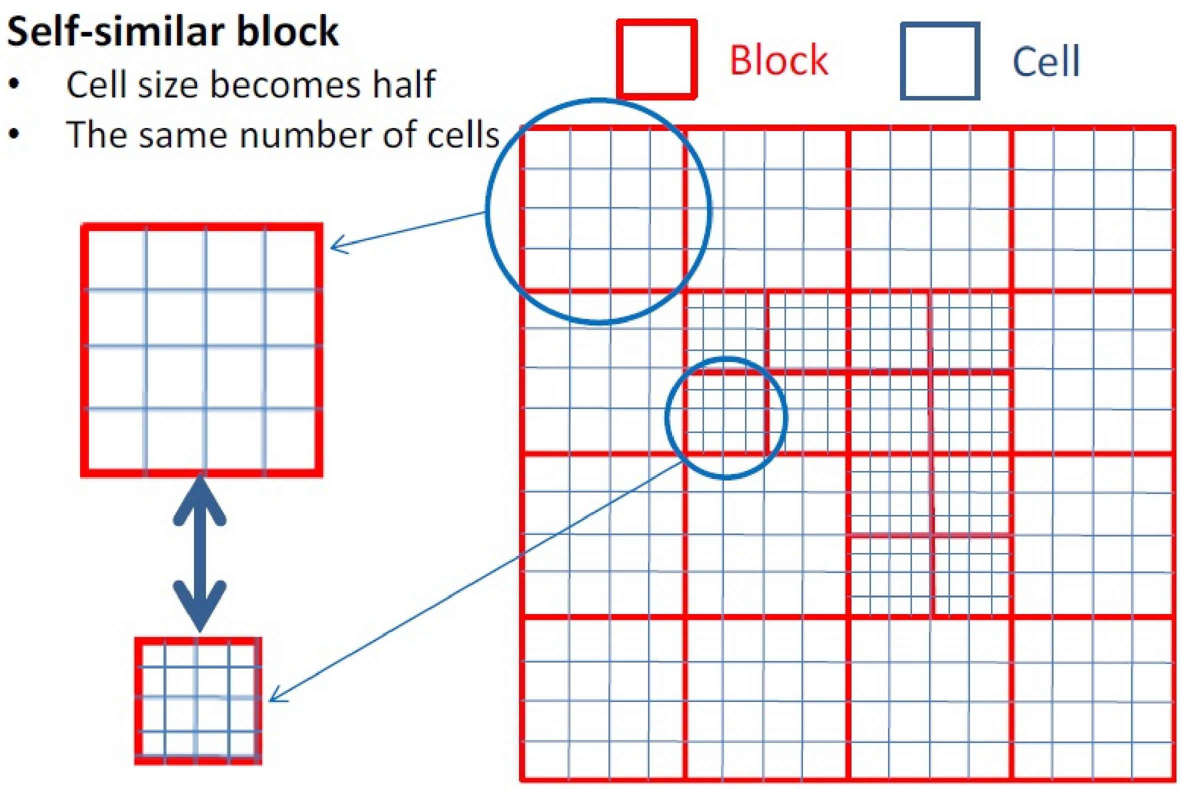


Figure 6: Example of block configuration in two-dimensional domain.

A simple example is shown in Figure (6) in which each block has (4 x 4) cells and the simulation domain consists of (4 x 4) base- blocks. We call a b lock with the coarsest cells the base-block. It should be noted that the number of cells in the refined block is the same as that in base-block because of self-similarity although the cell size becomes half. The number of cells in each block and the number of base -block consisting of the whole simulation domain can be initially set as input parameters.

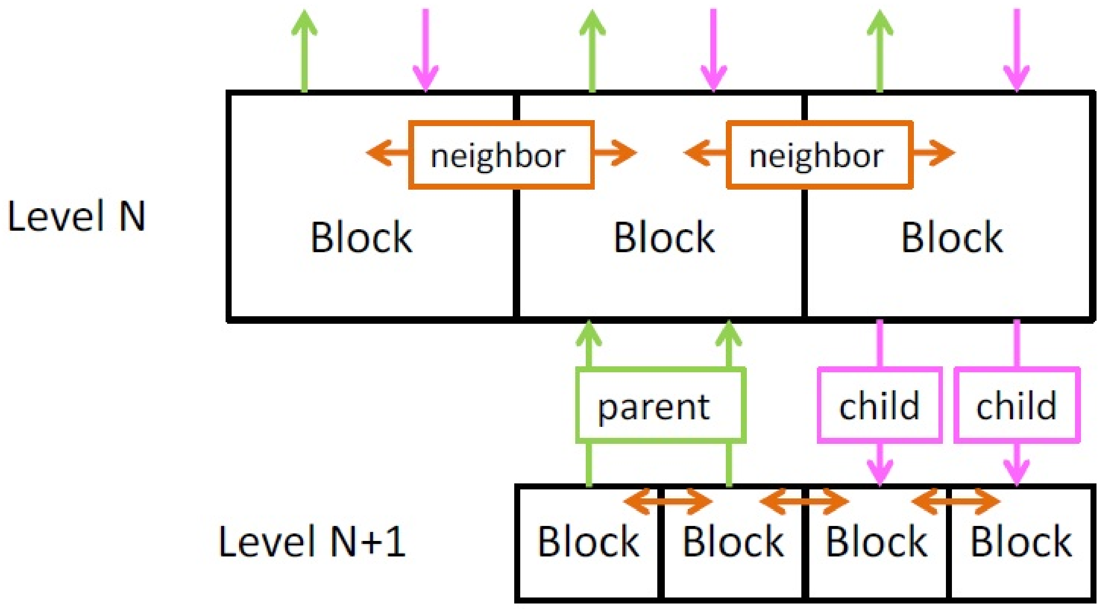


Figure 7: Example of block configuration in two-dimensional domain.

Figure (7) shows the block connections in the one-dimensional FTT structure. Each block has information about the level of refinement given in the FTT structure and the physical quantities defined at each cell in the block. Every b lock has three kinds of pointer wh ich refer to neighbor, child, and parent blocks determined in the FTT structure. The neighbor pointer configures connections to adjacent blocks in the same hierarchical level. The child pointers and the parent pointer configure connections to blocks with fine and coarse cells located in the same region in different hierarchical levels, respectively. The parent block remains when child blocks are generated. In three-dimensional model, the physical quantities in each cell in the child blocks are obtained by the linear interpolation of those of eight parent-level cells which are most closely located around the corresponding child cell.

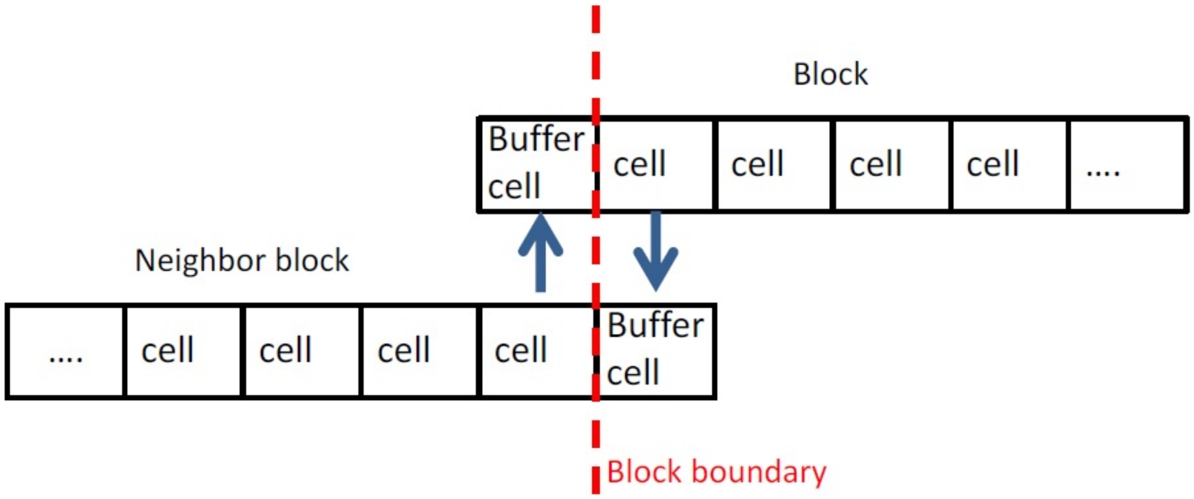


Figure 8: Data exchange between buffer cells in the base-block level

Numerical connection to neighboring blocks is configured by a buffer region. As shown in Figure (8) and (9), each block has a buffer region at the edges of the physical do main of the block. The buffer region consists of additional cells which are necessary in updating the physical quantities in the physical domain by the finite difference method. In the base-block level, as shown in (a) o f Figure 3, the

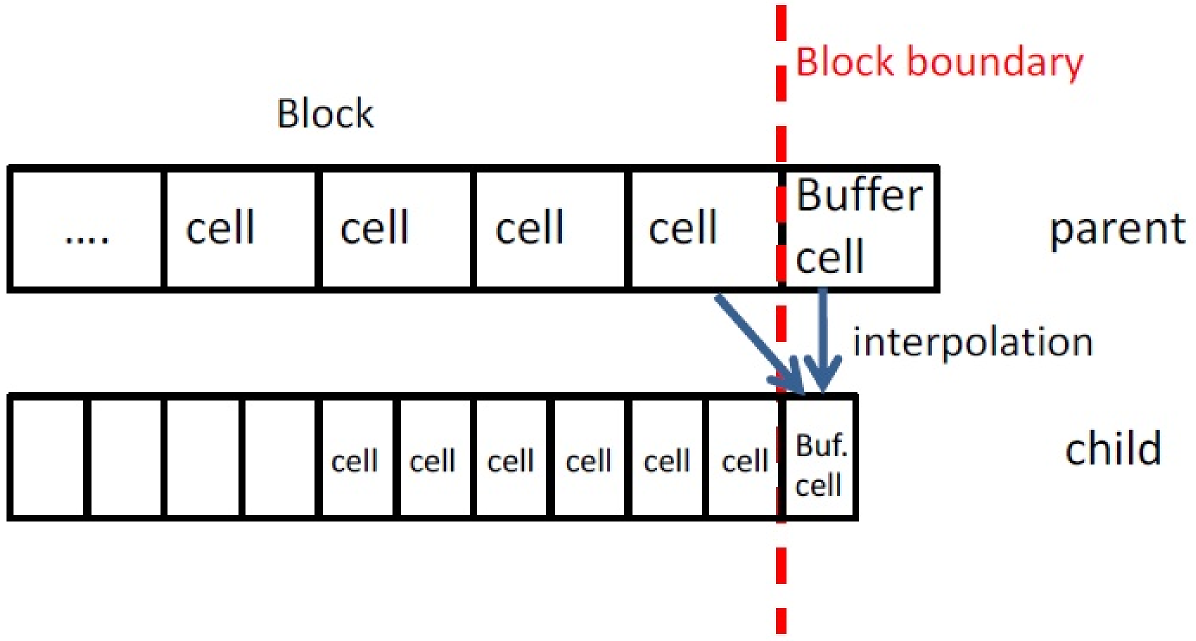


Figure 9: Data copy to buffer cell from the parent block by interpolation.

physical values at the boundary of the adjacent block are simply copied to the cells in the buffer region. Although one cell is assigned for the buffer region in Figure (8), the number of the buffer cells can be arbitrarily set by the users depending on the requirement of the fin ite difference method used in the corresponding simulation solver. In the child block, as shown in Figure Figure (9), a buffer region will also be attached at the edge of the physical region and the physical quantities in the buffer cell are inherited from the base-block by the linear interpolation as explained above. By making use of the buffer region, we can treat each block as an independent unit even if blocks in different level of hierarchy face to each other at the block boundary. As stated earlier, the hierarchical relation of all blocks is to be managed in the FFT structure, which allows us to incorporate the AMR treatment in simulation with a high degree of flexibility.

Since the framework can handle the hierarchical relation among the blocks with the FTT structure, what the users (i.e. us) basically have to prepare is the outer boundary condition for the entire simulation space, the main routine for calculation in each block, and a criterion for the cell refinement. Depending on the problem specification, we need to prepare these three parts accordingly.

References

Ahn, H. and Shashkov, M. (2009). Adaptive moment-of-fluid method. *Journal of Computational Physics*, 228(8):2792–2821.

Aniszewski, W., MÃl'nard, T., and Marek, M. (2014). Volume of fluid (vof) type advection methods in two-phase flow: A comparative study. *Computers and Fluids*, 97:52–73.

Ashgriz, N. and Poo, J. Y. (1991). FLAIR: Flux line-segment model for advection and interface reconstruction. *Journal of Computational Physics*, 93(2):449–468.

Aulisa, E., Manservisi, S., Scardovelli, R., and Zaleski, S. (2007). Interface reconstruction with least-squares fit and split advection in three-dimensional Cartesian geometry. *Journal of Computational Physics*, 225(2):2301–2319.

Berger, M. J. and Oliger, J. (1984). An adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53:484–512.

Brackbill, J. U., Kothe, D. B., and Zemach, C. (1992). A continuum method for modeling surface tension. *Journal of Computational Physics*, 100(2):335–354.

Ceniceros, H. D., Nós, R. L., and Roma, A. M. (2010). Three-dimensional, fully adaptive simulations of phase-field fluid models. *Journal of Computational Physics*, 229(17):6135–6155.

Chan, R. K. and Street, R. L. (1970). A computer study of finite-amplitude water waves. *Journal of Computational Physics*, 6(1):68–94. Cited By :183.

Chang, Y. C., Hou, T. Y., Merriman, B., and Osher, S. (1996). A level set formulation of Eulerian interface capturing methods for incompressible fluid flows. *Journal of Computational Physics*, 124(2):449–464.

Chapman, R. and Plesset, M. (1972). Nonlinear effects in the collapse of a nearly spherical cavity in a liquid. *J Basic Eng Trans ASME*, 94 Ser D(1):142–146. Cited By :28.

Chen, S., Johnson, D. B., Raad, P. E., and Fadda, D. (1997). The surface marker and micro cell method. *International Journal for Numerical Methods in Fluids*, 25(7):749–778.

Daly, B. J. (1969a). Numerical study of the effect of surface tension on interface instability. *Physics of Fluids*, 12(7):1340–1354. Cited By :65.

Daly, B. J. (1969b). A technique for including surface tension effects in hydrodynamic calculations. *Journal of Computational Physics*, 4(1):97–117. Cited By :61.

De Bar, R. (1974). Fundamentals of the KRAKEN code. *Technical Report*.

DeZeeuw, D. and Powell, K. (1993). An adaptively refined cartesian mesh solver for the euler equations. *Journal of Computational Physics*, 104(1):56–68.

Diot, S. and François, M. M. (2016). An interface reconstruction method based on an analytical formula for 3D arbitrary convex cells. *Journal of Computational Physics*, 305:63–74.

Dyadechko, V. and Shashkov, M. (2005). Interface tracking capabilities of the inter-gamma differencing scheme. Technical report, Los Alamos National Laboratory.

Enright, D., Fedkiw, R., Ferziger, J., and Mitchell, I. (2002). A hybrid particle level set method for improved interface capturing. *Journal of Computational Physics*, 183(1):83–116.

Fedkiw, R. P., Aslam, T., Merriman, B., and Osher, S. (1999). A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method). *Journal of Computational Physics*, 152(2):457–492.

Foote, G. (1973). A numerical method for studying liquid drop behavior: Simple oscillation. *Journal of Computational Physics*, 11(4):507–530. Cited By :50.

Foote, G. B. (1975). Water drop rebound problem: Dynamics of collision. Cited By :46.

Fuster, D., Bagué, A., Boeck, T., Le Moyne, L., Leboissetier, A., Popinet, S., Ray, P., Scardovelli, R., and Zaleski, S. (2009). Simulation of primary atomization with an octree adaptive mesh refinement and VOF method. *International Journal of Multiphase Flow*, 35(6):550–565.

Gaudlitz, D. and Adams, N. (2008). On improving mass-conservation properties of the hybrid particle-level-set method. *Computers and Fluids*, 37(10):1320–1331.

Glimm, J., Grove, J. W., Li, X. L., Oh, W., and Sharp, D. H. (2001). A Critical Analysis of Rayleigh-Taylor Growth Rates. *Journal of Computational Physics*, 169(2):652–677.

Gopala, V. R. and van Wachem, B. G. M. (2008). Volume of fluid methods for immiscible-fluid and free-surface flows. *Chemical Engineering Journal*, 141(1-3):204–221.

Harlow, F. H. and Shannon, J. P. (1967). The splash of a liquid drop. *Journal of Applied Physics*, 38(10):3855–3866. Cited By :219.

Harlow, F. H. and Welch, J. E. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids*, 8(12):2182–2189. Cited By :3037.

Harlow, F. H. and Welch, J. E. (1966). Numerical study of large-amplitude free-surface motions. *Physics of Fluids*, 9(5):842–851. Cited By :113.

Hirt, C. W. and Nichols, B. D. (1981). Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39(1):201–225.

Jacqmin, D. (1999). Calculation of Two-Phase Navier-Stokes Flows Using Phase-Field Modeling. *Journal of Computational Physics*, 155(1):96–127.

Jamet, D., Lebaigue, O., Coutris, N., and Delhaye, J. M. (2001). The Second Gradient Method for the Direct Numerical Simulation of Liquid-Vapor Flows with Phase Change. *Journal of Computational Physics*, 169(2):624–651.

Jasak, H. and Weller, H. (1995). Interface tracking capabilities of the inter-gamma differencing scheme. Technical report, Imperial College of Science, Technology and Medicine.

Ji, H., Lien, F.-S., and Yee, E. (2010). A new adaptive mesh refinement data structure with an application to detonation. *Journal of Computational Physics*, 229(23):8981–8993.

Kawano, A. (2016). A simple volume-of-fluid reconstruction method for three-dimensional two-phase flows. *Computers and Fluids*, 134-135:130–145.

Khokhlov, A. (1998). Fully threaded tree algorithms for adaptive refinement fluid dynamics simulations. *Journal of Computational Physics*, 143(2):519–543.

Kobayashi, R. (1992). Simulations of three dimensional dendrites. *Pattern Formation in Complex Dissipative Systems*, pages 121–128.

Kobayashi, R. (1993). Modeling and numerical simulations of dendritic crystal growth. *Physica D: Nonlinear Phenomena*, 63(3-4):410–423.

Kolomenskiy, D., Nave, J.-C., and Schneider, K. (2016). Adaptive Gradient-Augmented Level Set Method with Multiresolution Error Estimation. *Journal of Scientific Computing*, 66(1):116–140.

Lafaurie, B., Nardone, C., Scardovelli, R., Zaleski, S., and Zanetti, G. (1994). Modelling Merging and Fragmentation in Multiphase Flows with SURFER. *Journal of Computational Physics*, 113(1):134–147.

Laurmaa, V., Picasso, M., and Steiner, G. (2016). An octree-based adaptive semi-Lagrangian VOF approach for simulating the displacement of free surfaces. *Computers and Fluids*, 131:190–204.

Lee, L. and Leveque, R. J. (2003). An immersed interface method for incompressible Navier-Stokes equations. *SIAM Journal on Scientific Computing*, 25(3):832–856.

Liu, H., Krishnan, S., Marella, S., and Udaykumar, H. S. (2005). Sharp interface Cartesian grid method II: A technique for simulating droplet interactions with surfaces of arbitrary shape. *Journal of Computational Physics*, 210(1):32–54.

Losasso, F., Fedkiw, R., and Osher, S. (2006). Spatially adaptive techniques for level set methods and incompressible flow. *Computers and Fluids*, 35(10):995–1010.

MacNeice, P., Olson, K. M., Mobarry, C., de Fainchtein, R., and Packer, C. (2000). PARAMESH: A parallel adaptive mesh refinement community toolkit. *Computer Physics Communications*, 126:330–354.

Marella, S., Krishnan, S., Liu, H., and Udaykumar, H. S. (2005). Sharp interface Cartesian grid method I: An easily implemented technique for 3D moving boundary computations. *Journal of Computational Physics*, 210(1):1–31.

Mitchell, T. M. and Hammitt, F. G. (1973). Asymmetric cavitation bubble collapse. *Journal of Fluids Engineering, Transactions of the ASME*, 95 Ser I(1):29–37. Cited By :29.

Noh, W. F. and Woodward, P. (1976). SLIC (simple line interface calculation). *Lecture Notes in Physics*, 59:330–340.

Osher, S. and Sethian, J. A. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49.

Peskin, C. S. (1977). Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3):220–252.

Pivello, M. R., Villar, M. M., Serfaty, R., Roma, A. M., and Silveira-Neto, A. (2014). A fully adaptive front tracking method for the simulation of two phase flows. *International Journal of Multiphase Flow*, 58:72–82.

Popinet, S. (2003). Gerris: A tree-based adaptive solver for the incompressible Euler equations in complex geometries. *Journal of Computational Physics*, 190(2):572–600.

Popinet, S. (2009). An accurate adaptive solver for surface-tension-driven interfacial flows. *Journal of Computational Physics*, 228(16):5838–5866.

Powell, K. (1998). Solution of the Euler equations on solution-adaptive Cartesian grids. *In Computational Fluid Dynamics Reviews 1998*, 1:65–92.

Pulliam, T. H. and Steger, J. L. (1980). Implicit finite-difference simulations of three-dimensional compressible flow. *AIAA journal*, 18(2):159–167.

Quirk, J. J. (1994). An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies. *Computers and Fluids*, 23(1):125–142.

Renardy, Y. and Renardy, M. (2002). PROST: A parabolic reconstruction of surface tension for the volume-of-fluid method. *Journal of Computational Physics*, 183(2):400–421.

Rider, W. J. and Kothe, D. B. (1998). Reconstructing volume tracking. *Journal of Computational Physics*, 141(2):112–152. Cited By :822.

Rudman, M. (1997). Volume-tracking methods for interfacial flow calculations. *International Journal for Numerical Methods in Fluids*, 24(7):671–691.

Scardovelli, R. and Zaleski, S. (2000). Analytical Relations Connecting Linear Interfaces and Volume Fractions in Rectangular Grids. *Journal of Computational Physics*, 164(1):228–237.

Sussman, M. (1994). A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114(1):146–159.

Sussman, M., Almgren, A. S., Bell, J. B., Colella, P., Howell, L. H., and Welcome, M. L. (1999). An Adaptive Level Set Approach for Incompressible Two-Phase Flows. *Journal of Computational Physics*, 148(1):81–124.

Sussman, M. and Fatemi, E. (1999). Efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM Journal on Scientific Computing*, 20(4):1165–1191.

Sussman, M., Fatemi, E., Smereka, P., and Osher, S. (1998). An improved level set method for incompressible two-phase flows. *Computers and Fluids*, 27(5-6):663–680.

Sussman, M. and Puckett, E. G. (2000). A Coupled Level Set and Volume-of-Fluid Method for Computing 3D and Axisymmetric Incompressible Two-Phase Flows. *Journal of Computational Physics*, 162(2):301–337.

Sussman, M. and Smereka, P. (1997). Axisymmetric free boundary problems. *Journal of Fluid Mechanics*, 341:269–294.

Takewaki, H., Nishiguchi, A., and Yabe, T. (1985). Cubic interpolated pseudo-particle method (CIP) for solving hyperbolic-type equations. *Journal of Computational Physics*, 61(2):261–268.

Tan, H. (2016). An adaptive mesh refinement based flow simulation for free-surfaces in thermal inkjet technology. *International Journal of Multiphase Flow*, 82:1–16.

Thomas, P. D. and Lombard, C. K. (1979). Geometric conservation law and its application to flow computations on moving grids. *AIAA journal*, 17(10):1030–1037.

Tryggvason, G. and Aref, H. (1983). Numerical experiments on Hele Shaw flow with a sharp interface. *Journal of Fluid Mechanics*, 136:1–30.

Ubbink, O. (1997). *Numerical prediction of two fluid systems with sharp interfaces*. PhD thesis, Imperial College London.

Udaykumar, H. S., Mittal, R., Rampunggoon, P., and Khanna, A. (2001). A sharp interface Cartesian grid method for simulating flows with complex moving boundaries. *Journal of Computational Physics*, 174(1):345–380.

Unverdi, S. O. and Tryggvason, G. (1992). A front-tracking method for viscous, incompressible, multi-fluid flows. *Journal of Computational Physics*, 100(1):25–37.

Van Wachem, B. G. M. and Schouten, J. C. (2002). Experimental validation of 3-D Lagrangian VOF model: Bubble shape and rise velocity. *AIChE Journal*, 48(12):2744–2753.

Vinokur, M. (1989). An analysis of finite-difference and finite-volume formulations of conservation laws. *Journal of Computational Physics*, 81(1):1–52.

Visbal, M. and Gaitonde, D. (2002). On the use of higher-order finite-difference schemes on curvilinear and deforming meshes. *Journal of Computational Physics*, 181(1):155–185.

Wang, Z., Yang, J., and Stern, F. (2009). An improved particle correction procedure for the particle level set method. *Journal of Computational Physics*, 228(16):5819–5837.

Xiao, F., Honma, Y., and Kono, T. (2005). A simple algebraic interface capturing scheme using hyperbolic tangent function. *International Journal for Numerical Methods in Fluids*, 48(9):1023–1040.

Xiao, F., Ii, S., and Chen, C. (2011). Revisit to the {THINC} scheme: A simple algebraic {VOF} algorithm. *Journal of Computational Physics*, 230(19):7086–7092.

Yabe, T., Xiao, F., and Utsumi, T. (2001). The Constrained Interpolation Profile Method for Multiphase Analysis. *Journal of Computational Physics*, 169(2):556–593.

Yang, Y. and Udaykumar, H. S. (2005). Sharp interface Cartesian grid method III: Solidification of pure materials and binary solutions. *Journal of Computational Physics*, 210(1):55–74.

Yokoi, K. (2007). Efficient implementation of thinc scheme: A simple and practical smoothed vof algorithm. *Journal of Computational Physics*, 226(2):1985–2002.

Youngs, D. L. (1982). Time-dependent multi-material flow with large fluid distortion. *Numerical Methods for Fluid Dynamics*, pages 273–285.