Spring 11-13-2019

A Study of Methods for Improving Volume Conservation in Incompressible Two-Phase Flow on Unstructured Meshes

Meredith Katherine Purser
University of Colorado at Boulder, mkmpurser@gmail.com

A Study of Methods for Improving Volume Conservation in Incompressible Two-Phase Flow on Unstructured Meshes

by

Meredith K. Purser

S.B., Massachusetts Institute of Technology, 2006

M.E., Pennsylvania State University, 2009

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Aerospace Engineering Sciences

2019

This thesis entitled:
A Study of Methods for Improving Volume Conservation in Incompressible Two-Phase Flow on Unstructured Meshes
written by Meredith K. Purser
has been approved for the Department of Aerospace Engineering Sciences

_____

Kenneth Jansen

_____

Prof. John Farnsworth

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Purser, Meredith K. (Ph.D., Aerospace Engineering)

A Study of Methods for Improving Volume Conservation in Incompressible Two-Phase Flow on Unstructured Meshes

Thesis directed by Prof. Kenneth Jansen

Computational modeling of incompressible two-phase flows requires an efficient means of tracking the interface between phases. An ideal method would directly provide information about the interface's location and curvature, would be strictly volume conserving, and would work for parallel computing on unstructured meshes. For incompressible two-phase flows, there are several popular interface tracking schemes, but each has significant shortcomings. For example, while level set schemes directly provide the location and curvature of the interface, they do not strictly conserve volume. Conversely, volume of fluid methods strictly conserve volume, but they do not directly provide the interface's location or curvature. Coupled level set volume of fluid (CLSVOF) methods mitigate some of the weaknesses of both level set and volume of fluid methods while preserving many of their strengths, but CLSVOF methods face challenges yet to be resolved for unstructured meshes.

This work first examines the challenges in implementing a CLSVOF method on unstructured meshes for parallel computing. It does not present a complete CLSVOF method, but it lays the foundation and makes recommendations for future work. This work also examines several variations of level set methods to develop an understanding of the trade-offs inherent in the methods (e.g., preservation of curvature versus volume conservation, and computational cost versus volume conservation). The developments of this work are implemented in Parallel Hierarchic Adaptive Stabilized Transient Analysis (PHASTA) software, a proven finite element flow solver, and the level set variations are tested on several canonical problems.

This work also examines the use of interface-based adaptive mesh refinement (AMR) to improve interface tracking and volume conservation. Proximity to the interface and/or the curvature of the level set field can be used to define areas of mesh refinement, thereby reducing the discretization errors near the interface without incurring unmanageable computational expense. This approach to AMR is demonstrated on a dam break problem and an annular flow simulation. It is also paired with the level set variations to offer a more complete picture of the options available.

Dedication

To my dad who first inspired me to pursue a Ph.D. and to my mom who has ceaselessly encouraged my love of math, science, and engineering.

Acknowledgements

My deepest thanks go to Prof. Kenneth Jansen, not only for his support, encouragement, and patience, but also for his instruction and guidance. In addition to teaching me a great deal about computational fluid dynamics, he has helped me to hone my critical thinking and problem solving skills.

My thanks also go to Prof. Brian Argrow, Prof. John Evans, Prof. John Farnsworth, Prof. Peter Hamlington, and Prof. Ryan Starkey for sharing their time and their insights.

I am grateful to all those in the PHASTA research group at Boulder who have helped me along the way: Ben Matthews, Dr. Kedar Chitale, Dr. Michel Rasquin, Kyle Woolwine, and Ryan Skinner. If it were not for them, this journey would have been lonelier, more challenging, and a lot less fun.

Finally, my gratitude goes to my husband, Ben, for all of his patience and encouragement as we both worked to finish our degrees, and to all my family and friends who have done so much to support and motivate me.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

1.1 Background

Two-phase flows are found in nature in phenomena like ocean surface waves or rain, in commercial applications such as ink jet printers, and in industrial applications including phase-change heat exchangers and nuclear reactors. The commonality and diversity of two-phase flows mean that it is frequently important to be able to model two-phase flows well. Of particular importance in modeling two-phase flows is capturing the behavior of the interface between the phases. In some flows, the interface dynamics can be quite complex, including merging interfaces that accompany breaking waves and bubble/droplet formation.

One two-phase flow with such interface dynamics is annular flow, which is common in industrial applications. Annular flow, which is shown in figure 1.1, is a flow parallel to the axis of a pipe where a liquid film develops on the wall and the center of the pipe is filled with a higher speed gas core. The gas core entrains droplets, the liquid film entrains bubbles, and the interface between the film and the core is wavy with complex dynamics, including breaking waves. This complexity makes annular flow a challenging flow on which to evaluate the performance of interface tracking schemes.

Additionally, because of annular flow's complexity and importance, there has long been an effort to develop closure models and engineering tools that would allow accurate simulation of annular flow to be part of the engineering design process. Of existing closure models for annular flow, many are based on empirical data [27, 26, 83, 40, 91, 19], and some

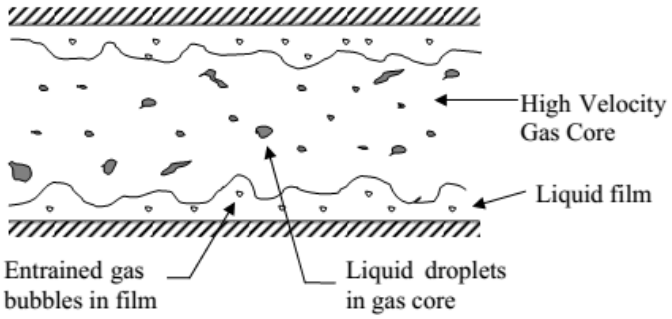


Figure 1.1: The gas core contains entrained droplests while the liquid film contains entrained bubbles. The interface between the film and the gas core is wavy. Image created by J. Rodriguez [65]; used with permission.

focus on the transport process at the interface [75, 41, 44, 45]. This means that an accurate characterization of the interface is necessary to support good closure models; however, the interface cannot be fully characterized through experimental means, leaving direct numerical simulation as the only viable option to generate the needed information [65]. While there have been some useful simulations of annular flow [22, 65], there is still a need for more simulation data. This means that simulations which have the primary purpose of evaluating the interface tracking scheme can also potentially be used for the generation of data that could ultimately be used in the development of an improved closure model.

The focus of this work is neither the development of a new closure model for annular flow nor the production of a large quantity of annular flow simulation data, but rather a study of interface tracking schemes that will be evaluated through the simulation of annular flow as well as simulations of several canonical test problems. This study seeks to improve interface tracking performance and evaluate the costs and benefits of multiple competing schemes. Interface tracking schemes will be evaluated based on their ease of implementation, computational cost, parallelizability, conservation of volume, and preservation of interface curvature.

1.2 State of the Art and Contribution of This Thesis

As shown by Tryggvason et al. [86] and Lakehal et al. [46], interface tracking methods broadly fall into four categories: front capturing, front tracking, boundary-fitted grids, and Lagrangian approaches. Both front capturing and front tracking schemes solve the flow on a stationary mesh; front capturing schemes then use a marker function to resolve the interface while front tracking schemes resolve the interface on a lower order mesh. Boundary-fitted schemes initialize and maintain grids in which the interface coincides with a coordinate in the grid coordinate system (i.e., grid edges in two dimensions, or faces in three dimensions, to continuously maintain alignment with the interface between fluid phases). In Lagrangian schemes, the mesh moves with the fluid everywhere in the domain, including at the interface. Neither boundary-fitted schemes nor Lagrangian schemes are well suited to flows with large topological changes in the interface (e.g., breaking waves and particle entrainment). Although all four categories of schemes have been developed successfully, front capturing has produced the broadest range of methods.

Front capturing schemes include several popular methods. Among them, some of the most popular are volume of fluid (VOF) methods, level set methods, and methods directly derived from either or both the level set and VOF methods. A common method derived from VOF methods is the moment of fluid (MOF) method (see, for example, [2, 71, 37]). Two methods derived from both level set and VOF methods are coupled level set volume of fluid (CLSVOF) methods and coupled level set moment of fluid (CLSMOF) methods. These methods have been used successfully on a variety of complex flow problems, however each of these methods (VOF, level set, MOF, CLSVOF, and CLSMOF) has at least one significant disadvantage. These methods and their general strengths and weaknesses are discussed in detail in the following subsections.

Front tracking schemes have been extensively developed by Tryggvason et al. [89, 86, 53, 85] and applied to several problems, but the complexity of maintaining an adequate

interface mesh and transferring information between the interface mesh and problem domain means that they do not have the broad popularity of front capturing schemes. Boundary-fitted grids have been shown to be quite accurate for problems with simple interfaces (e.g., Ryskin and Leal's study of bouyancy-driven bubbles and bubble deformation [68, 69], and studies of non-breaking waves by Fulgosi et al. [23] and Kurose et al. [42]), but the method does not readily handle large interface distortions or merging interfaces, making it unsuitable to use in complex flows. When Lagrangian front tracking methods have been used, they have often been arbitrary Lagrangian-Eulerian implementations (e.g., particulate flow studies by Hu et al. [30, 31] and Patankar and Joseph [61]). Recently, Langrangian schemes have been expanded to include hybrid methods using one or more front capturing schemes in order to accommodate more complex interface motion (e.g., flows with both fluid-solid interfaces and fluid-fluid interfaces as examined by Cruchaga et al. [16], Akin et al. [4], and Akkerman et al. [5]). However, like the boundary-fitted methods, Lagrangian schemes poorly handle merging interfaces and so are ill-suited to flows with complex interface dynamics. Accordingly, this work focuses on front capturing schemes.

1.2.1 Volume of Fluid and Moment of Fluid Methods

The VOF method was first introduced by Hirt and Nichols [28]. In their formulation, the volume fraction, here denoted F, is a cell-centered marker function that takes on a value of 1 in one phase (usually the liquid phase in a liquid-gas flow), a value of 0 in the other phase, and a fractional value equal to the portion of the cell occupied by the first phase in cells containing the interface. F is advected with the flow, and the interface is inferred using F and its gradient. Hirt and Nichols [28] and other early VOF implementations used a piecewise constant interface reconstruction such as Noh and Woodward's simple line interface calculation [57]. Subsequent developments in the VOF method (see [64, 25, 62, 49] for a small sampling) have improved the accuracy of both the advection algorithms and the interface construction algorithms.

VOF methods are popular because they can be strictly volume conserving and less memory intensive than related marker functions (see Scardovelli and Zaleski's [70] review of interface tracking methods). While VOF methods can handle merging interfaces (e.g., in cavitation simulations by Roohi et al. [67]), the need to infer the interface from the volume fraction means that it does not do so easily. Additionally, because the interface is generally piece-wise linear and allowed to be discontinuous across cell boundaries, calculating the curvature of the interface (which is needed for surface tension) can be quite inaccurate and lead to non-physical flows [12]. Furthermore, many of the more accurate VOF advection and interface construction schemes [64, 62] are not naturally suited to unstructured meshes, limiting the cases in which they can be applied.

A direct descendant of the VOF method is the MOF method. Dyadechko and Shashkov [17, 18] developed an interface reconstruction algorithm that uses the center of mass in each cell, as well as the volume fraction field, to reconstruct the interface. Interface reconstruction is done in such a way that the volume fraction F is exactly preserved and the movement of the center of mass is minimized. Ahn et al. [3] then developed a moment advection scheme to support the MOF interface reconstruction. The MOF method offers two key advantages over the VOF method: the MOF method can be expanded to handle more than two materials [1, 71], and it can be adapted to handle filaments where there are two interfaces within a single cell [38]. However, the MOF method still shares some limitations of the VOF method: calculating the curvature of the interface can still be challenging, as is handling merging interfaces. Furthermore, the more complicated moment advection scheme increases the challenges of implementing MOF for use on unstructured meshes.

1.2.2 Level Set Method

The level set method is based on the front-propagating scheme developed by Osher and Sethian [60]. In the level set method, the interface is represented by the zeroth level set. The level set itself is a signed distance field, here denoted by ϕ, the values of which are

stored nodally. At the interface, $\phi = 0$, and ϕ takes on positive values in one phase (usually the liquid phase) and negative values in the other. The level set, then, is a smooth function that not only directly gives the interface position, but by being spatially differentiable also makes computation of curvature and the handling of merging and breaking interfaces straightforward. The level set is advected with the flow, but velocity gradients in the flow can distort the level set so that it is no longer a true distance field. To correct that, the level set must be periodically "redistanced" or restored to being a true distance field. As a result of spatial discretization, the redistancing procedure tends to slightly move the interface, leading to lack of volume conservation. While a correction can be applied to the redistancing step [78, 79, 65, 66], this is a global correction that tends to lead to the loss of high curvature regions.

As highlighted in several reviews [72, 73, 58, 88] of level set methods, level sets are also a popular tool in image processing, particularly image segmentation. In image segmentation, a level set contour field is initialized (for example through the manual selection of points [15, 6] or by the use of a predefined geometric contour [87, 21]). The level set field is then evolved to find the lowest potential of an energy function (e.g., the Mumford-Shah functional [52]). The contours of the final field match the features of interest in the image. This type of level set application has found popular success, especially in medical image segmentation [59, 39, 29].

The popularity of level set methods is driven by their ability to handle merging and breaking interfaces, making them suitable to a wide variety of flow simulations as well as applications outside of fluid mechanics, including crystal growth and image processing [73]. Additionally, level set methods are easily implemented on both structured (as in Sussman et al.'s work [82, 80]) and unstructured meshes (as in the work of Rodriguez, et al. [66]). However, despite efforts to improve the level set's volume conservation performance [78, 79, 46, 65, 66], its lack of strict volume conservation remains its greatest weakness and is a principle driver behind the development of hybrid methods.

1.2.3 Hybrid Methods

The CLSVOF and CLSMOF methods couple the level set method with the VOF method and the MOF method, respectively. In the CLSVOF method as developed by Sussman [81, 77], both the level set and volume fraction are advected with the flow. At each time step, the interface is reconstructed so that in cells containing the interface (as defined by the level set field), the piece-wise linear interface has the normal that is given by the values of the level set ϕ and the location within the cell that preserves the volume fraction F. The level set field is then re-established within some distance of the interface. This re-establishing of the level set takes the place of the redistancing that occurs in the level set method. Away from the interface, the level set is permitted to not be a distance field, because away from the interface, only sign of the level set (not its magnitude) is used to establish fluid properties. Similarly, in the CLSMOF method developed by Jemison et al. [36], the interface slope as determined by the level set is used to accelerate the convergence of the MOF interface reconstruction.

Since its introduction, the CLSVOF method has become popular for solving axisymmetric problems (e.g., buoyant bubble rise [12], liquid drop impact [63], and droplet formation [13]) and other problems suited to a two-dimensional Cartesian grid (such as wave breaking in a channel [90]). It has similarly been demonstrated successfully on a number of three-dimensional problems with Cartesian grids (e.g., droplet coalescence and breakup [43] and droplet splashing [95]). The regularity of the Cartesian grids makes both the advection of the volume fraction and the re-establishing of the level set straightforward operations, meaning that CLSVOF is well-suited to structured grids. However, there has been some work to implement CLSVOF on unstructured meshes. Because CLSVOF is not naturally suited to unstructured meshes, using it on such a mesh requires additional complications; both Yang et al. [94] and Cao et al. [11, 10] introduced arbitrary Lagrangian-Eulerian approaches to the two-dimensional advection of the volume fraction.

The literature is clear that for simulations on structured grids, the CLSVOF method mitigates the weaknesses of both the VOF and level set methods while preserving their strengths. However, it is not clear that CLSVOF can be extended to unstructured three-dimensional meshes without introducing complexities (e.g., additional meshes) that would result in unacceptably high computational costs.

1.2.4 Contribution of this Thesis

Parallel computing on unstructured meshes is an area of two-phase fluid mechanics where improvements in interface tracking would be beneficial. To that end, this thesis examines some of the challenges of implementing a CLSVOF method for unstructured meshes and parallel computing. It does not, however, present a successful new CLSVOF method.

Additionally, this thesis investigates the performance of several level set variants in order to determine the tradeoffs between computational costs and algorithmic performance, both in terms of volume conservation and preservation of curvature. Specifically, three redistancing variations are examined: redistancing without constraints, redistancing with a Sussman-type global volume constraint [78, 79, 65, 66], and redistancing where the nodes nearest the interface are constrained from moving. These three variations produce different results in terms of speed, cost, preservation of curvature, and volume conservation.

1.3 Thesis Outline

The remainder of this thesis is organized in the following way. Chapter 2 discusses the numerical methods for solving incompressible flow and the level set, as implemented in the Parallel Hierarchic Adaptive Stabilized Transient Analysis (PHASTA) software. Chapter 3 presents the work that was done to develop a CLSVOF method for parallel computing on unstructured grids; it discusses both the successes and failures of that effort and offers suggestions for future development work. Chapter 4 presents the studied level set variations and their performance on several canonical test problems. Chapter 5 covers the implementation

and performance of interface-based adaptive mesh refinement (AMR) for the level set variations. Finally, Chapter 6 presents conclusions and recommendations for future work.

Chapter 2

Numerical Method: Flow Solution and Level Set

PHASTA [32] is a parallel, hierarchic (higher order accurate, from the 2nd to 5th order accuracy, depending on function choice), unstructured, adaptive, stabilized finite element transient analysis flow solver both for incompressible and compressible 3-dimensional flows. All of the developments and applications in this thesis use the version of PHASTA for incompressible flows, which solves the incompressible Navier-Stokes equations. PHASTA's incompressible flow solver is a proven code and has been demonstrated on a variety of two-phase problems [55, 65, 8, 66, 7].

The following two sections (2.1 and 2.2) review solving the incompressible Navier-Stokes equations and the level set method, both of which were fully implemented in PHASTA prior to the developments of this thesis.

2.1 Flow Solution

2.1.1 Governing Equations

The strong form of the incompressible Navier-Stokes equations is given by:

Continuity:

$$u_{i,j} = 0 \tag{2.1}$$

Momentum:

$$\rho u_{i,t} + \rho u_j u_{i,j} = -\hat{p}_{,i} + \hat{\tau}_{ij,j} + \hat{f}_i \tag{2.2}$$

where u_i is i^{th} component of velocity, $\hat{p}$ is the pressure, and $\hat{f}_i$ represents body forces along the i^{th} coordinate. Following the continuum surface tension model of Brackbill et al. [9], the surface tension is computed as a local body force and included in $\hat{f}_i$. For the incompressible flow of a Newtonian fluid, the stress tensor is related to the strain rate tensor, S_{ij}, as:

$$\hat{\tau}_{ij} = 2\mu S_{ij} = \mu(u_{i,j} + u_{j,i}). \tag{2.3}$$

2.1.2 Finite Element Discretization

PHASTA's spatial and temporal discretization of the incompressible Navier-Stokes equations have been described in detail by Whiting [92], Nagrath [56], and Rodriguez [65]. The discussion given here follows theirs.

To derive the finite element discretization of the weak form of the equations of motion (2.1,2.2) it is necessary to introduce the discrete weight and solution function spaces. Let $\bar{\Omega} \subset \mathbf{R}^N$ represent the closure of the physical spatial domain (i.e., $\Omega \cup \Gamma$ where Γ is the boundary) in N dimensions; only $N = 3$ is considered here. The boundary is decomposed into portions with natural boundary conditions, Γ_h, and essential boundary conditions, Γ_g, i.e., $\Gamma = \Gamma_g \cup \Gamma_h$. In addition, $H^1(\Omega)$ represents the Sobolev space of functions with square-integrable values and derivatives on Ω.

Subsequently Ω is discretized into n_{el} finite elements, $\bar{\Omega}_e$. With this, the discrete trial solution and weight spaces for the semi-discrete formulation can be defined as:

$$\boldsymbol{\mathcal{S}}_h^k = \{\boldsymbol{v} | \boldsymbol{v}(\cdot,t) \in H^1(\Omega)^N, t \in [0,T], \boldsymbol{v}|_{x\in\bar{\Omega}_e} \in P_k(\bar{\Omega}_e)^N, \boldsymbol{v}(\cdot,t) = \boldsymbol{g} \text{ on } \Gamma_g\}, \tag{2.4}$$

$$\boldsymbol{\mathcal{W}}_h^k = \{\boldsymbol{w} | \boldsymbol{w}(\cdot,t) \in H^1(\Omega)^N, t \in [0,T], \boldsymbol{w}|_{x\in\bar{\Omega}_e} \in P_k(\bar{\Omega}_e)^N, \boldsymbol{w}(\cdot,t) = \boldsymbol{0} \text{ on } \Gamma_g\}, \tag{2.5}$$

$$\mathcal{P}_h^k = \{p | p(\cdot,t) \in H^1(\Omega), t \in [0,T], p|_{x\in\bar{\Omega}_e} \in P_k(\bar{\Omega}_e)\} \tag{2.6}$$

where $P_k(\bar{\Omega}_e)$ is the space of all polynomials defined on Ω^e, complete to order $k \geq 1$. The local approximation space, $P_k(\bar{\Omega}_e)$, is the same for both the velocity and pressure variables, which is possible due to the stabilized nature of the formulation shown below.

To generate the weak form of Eqs. (2.1 and 2.2), the left-hand sides are dotted with the weight functions, and the result is integrated over the domain. The pressure term, viscous stress term, and the continuity equation are all integrated by parts to eliminate second derivatives in the solution space. The residuals of the continuity and momentum equations are then added together, following the stabilized formulation by Taylor et al. [84]. The semi-discrete Galerkin finite element formulation is then given by:

Find $\boldsymbol{u} \in \boldsymbol{\mathcal{S}}_h^k$ and $p \in \mathcal{P}_h^k$ such that

$$B_G(w_i, q; u_i, p) = 0, \tag{2.7}$$

$$\begin{aligned} B_G(w_i, q; u_i, p) = \int_\Omega \{ & w_i \left(\dot{u}_i + u_j u_{i,j} - f_i \right) + w_{i,j} \left(-p\delta_{ij} + \tau_{ij} \right) - q_{,i} u_i \} dx \\ & + \int_{\Gamma_h} \left\{ w_i \left(p\delta_{in} - \tau_{in} \right) + q u_n \right\} ds \end{aligned} \tag{2.8}$$

for all $\boldsymbol{w} \in \boldsymbol{\mathcal{W}}_h^k$ and $q \in \mathcal{P}_h^k$. The boundary integral term arises from integration by parts and is only carried out over the portion of the domain with natural boundary conditions.

Since the Galerkin method is unstable for the equal-order interpolations given above, it is necessary to include additional stabilization terms:

Find $\boldsymbol{u} \in \boldsymbol{\mathcal{S}}_h^k$ and $p \in \mathcal{P}_h^k$ such that,

$$B(w_i, q; u_i, p) = 0, \tag{2.9}$$

$$\begin{aligned} B(w_i, q; u_i, p) = & B_G(w_i, q; u_i, p) \\ & + \sum_{e=1}^{n_{el}} \int_{\bar{\Omega}_e} \left\{ \tau_M (u_j w_{i,j} + q_{,i}) \mathcal{L}_i + \tau_C w_{i,i} u_{j,j} \right\} dx \\ & + \sum_{e=1}^{n_{el}} \int_{\bar{\Omega}_e} \left\{ w_i \overset{\Delta}{u}_j u_{i,j} + \overset{\Delta}{\tau} u_j w_{i,j} \overset{\Delta}{u}_k u_{i,k} \right\} dx \end{aligned} \tag{2.10}$$

for all $\boldsymbol{w} \in \boldsymbol{\mathcal{W}}_h^k$ and $q \in \mathcal{P}_h^k$. For simplicity, $\mathcal{L}_i$ represents the residual of the i^{th} momentum equation,

$$\mathcal{L}_i = \dot{u}_i + u_j u_{i,j} + p_{,i} - \tau_{ij,j} - f_i. \tag{2.11}$$

The second line in the stabilized formulation, Eq. (2.10), represents the typical streamline upwind Petrov-Galerkin stabilization added to the Galerkin formulation for the incompressible set of equations (e.g., by Franca and Frey [20]). The third line in Eq. (2.10) consists of a term introduced by Taylor et al. [84] to overcome the loss of momentum conservation caused by the stabilization of momentum in the continuity equation, and a stabilization term for this added advective term. In that line

$$\overset{\Delta}{u}_i = -\tau_M \mathcal{L}_i. \tag{2.12}$$

The stabilization parameters in Eqs. (2.10) and (2.12), for which Whiting and Jansen [93] give a detailed description, are given for momentum, continuity, and the new advective term by:

$$\tau_M = \frac{C}{\sqrt{\frac{c_1}{\delta t^2} + c_2 u_i g_{ij} u_j + c_3 \nu^2 g_{ij} g_{ij}}}, \tag{2.13}$$

$$\tau_C = \frac{1}{8\tau_M \mathbf{tr}(g_{ij})}, \tag{2.14}$$

$$\bar{\tau} = \frac{C}{\sqrt{c_2 \overset{\Delta}{u}_i g_{ij} \overset{\Delta}{u}_j}} \tag{2.15}$$

where the coeffiecents C, c_1, c_2, and c_3 are defined based on the one-dimensional linear advection-diffusion equation using a linear finite element basis and g_{ij} is the covariant metric tensor for mapping from global to parametric element coordinates.

In order to derive a discrete system of algebraic equations, the weight functions (w_i, q), the solution variables (u_i, p), and their time derivatives from Eq. (2.10) are expanded in terms of the finite element basis functions. Whiting [92] gives a detailed description of the basis functions in PHASTA. Evaluating the spatial integrals using Gaussian quadrature yields this system of first-order, nonlinear, differential-algebraic equations:

$$R_A(\mathbf{u}_i, \mathbf{u}_{i,t}, \mathbf{p}) = 0, \qquad A = 1...n_s \tag{2.16}$$

where R_A is the residual, A denotes the arbitrary values of the weight functions, and $\mathbf{u}_i$, $\mathbf{u}_{i,t}$, and $\mathbf{p}$ are the solution vectors in the discrete space.

2.1.3 Generalized-α Time Integrator

Within PHASTA, the incompressible Navier-Stokes equations are integrated in time using the generalized-α method, which has been described in detail by Whiting [92] and Jansen, et al. [35]. The generalized-α method is a predictor-multi-corrector algorithm in which a generic solution is advanced from time step n to time step $n+1$ through the these four equations:

$$y_{n+1} = y_n + \Delta t \dot{y}_n + \gamma \Delta t (\dot{y}_{n+1} - \dot{y}_n), \tag{2.17}$$

$$\dot{y}_{n+\alpha_m} = \lambda y_{n+\alpha_f}, \tag{2.18}$$

$$\dot{y}_{n+\alpha_m} = \dot{y}_n + \alpha_m (\dot{y}_{n+1} - \dot{y}_n), \tag{2.19}$$

$$y_{n+\alpha_f} = y_n + \alpha_f (y_{n+1} - y_n) \tag{2.20}$$

where, for second-order accuracy,

$$\gamma = \frac{1}{2} + \alpha_m - \alpha_f \qquad \alpha_m = \frac{1}{2}\left(\frac{3-\rho_\infty}{1+\rho_\infty}\right) \qquad \alpha_f = \frac{1}{1+\rho_\infty} \tag{2.21}$$

and ρ_∞ is a user-defined parameter; for example, $\rho_\infty = 0$ produces Gear's backward difference method [24], while $\rho_\infty = 1$ produces the midpoint rule.

Within PHASTA, the algorithm works by first predicting the solution at the next time step $(n+1)$ to be equal to the solution at the current time step (n):

$$u_{n+1} = u_n \qquad \dot{u}_{n+1} = \frac{\gamma - 1}{\gamma} \dot{u}_n \qquad p_{n+1} = p_n. \tag{2.22}$$

That prediction preserves second-order accuracy for the predictor-multi-corrector algorithm [92]. The first step in the correction process is to compute the velocity and acceleration at intermediate times for the i^{th} iteration:

$$u^{(i)}_{n+\alpha_f} = u_n + \alpha_f (u^{(i-1)}_{n+1} - u_n), \tag{2.23}$$

$$\dot{u}^{(i)}_{n+\alpha_m} = \dot{u}_n + \alpha_m (\dot{u}^{(i-1)}_n - \dot{u}_n). \tag{2.24}$$

These intermediate quantities can then be used to evaluate the residual, Eq. (2.16) for the i^{th} iteration. The result, however, is for small i likely to be far from the desired value of zero, meaning that the values resulting from Eqs. (2.23) and (2.24) should be improved. That improvement is found through linearizing $R^{(i)}$ about $\dot{u}^{(i)}_{n+1}$ and p^{i}_{n+1}, creating a matrix problem that can be solved for the changes in acceleration and pressure:

$$\begin{pmatrix} K^{(i)} & G^{(i)} \\ D^{(i)} & C^{(i)} \end{pmatrix} \begin{pmatrix} \delta\dot{u}^{(i)}_{n+1} \\ \delta p^{(i)}_{n+1} \end{pmatrix} = - \begin{pmatrix} R^{(i)}_{m} \\ R^{(i)}_{c} \end{pmatrix} \tag{2.25}$$

where $R^{(i)}_m$ and $R^{(i)}_c$ are the momentum and continuity components of the residual respectively, and $K^{(i)}, G^{(i)}, D^{(i)}$, and $C^{(i)}$ are the tangent matrices approximated by:

$$K^{(i)} \approx \frac{\partial R^{(i)}_m}{\partial \dot{u}^{(i)}_{n+1}} \qquad G^{(i)} \approx \frac{\partial R^{(i)}_m}{\partial p^{(i)}_{n+1}} \qquad D^{(i)} \approx \frac{\partial R^{(i)}_c}{\partial \dot{u}^{(i)}_{n+1}} \qquad C^{(i)} \approx \frac{\partial R^{(i)}_c}{\partial p^{(i)}_{n+1}}. \tag{2.26}$$

Following the solution of Eq. (2.25), the velocity, acceleration, and pressures solutions are updated:

$$u^{i+1}_{n+1} = u^{i}_{n+1} + \gamma\delta t\delta\dot{u}^{(i)}_{n+1}, \tag{2.27}$$

$$\dot{u}^{(i+1)}_{n+1} = \dot{u}^{(i)}_{n+1} + \delta\dot{u}^{(i)}_{n+1}, \tag{2.28}$$

$$p^{(i+1)}_{n+1} = p^{(i)}_{n+1} + \delta\dot{p}^{(i)}. \tag{2.29}$$

The correction process then returns to compute new intermediate time values for the velocity and acceleration (Eqs. (2.23) and (2.24)). The correction algorithm continues for a sufficient number of iterations to converge the non-linear equations.

2.2 Level Set

In level set methods, the level set is a signed distance field, here denoted by ϕ. The interface is defined by $\phi = 0$, and the two fluids are denominated by the sign of ϕ, with ϕ taking on positive values in one fluid (usually the liquid phase) and negative values in the other. As the scalar ϕ is advected with the flow, velocity gradients distort it so that it is no

longer a true distance field. To correct this, the level set field must be periodically reset, or redistanced. This redistancing is done using a second scalar, here denoted by d, and restores the level set to a true distance field.

2.2.1 Level Set: The First Scalar

The level set method implemented in PHASTA is based on that of Sussman [80, 79, 78] and Sethian [72, 74]. In it, the interface is modeled as the zero level set of the smooth signed distance field ϕ, which is advected with the flow:

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + \vec{u} \cdot \nabla \phi = 0. \tag{2.30}$$

Because ϕ is positive in one fluid (usually the liquid phase) and negative in the other, it can be used to determine the fluid properties. However, using a step change to evaluate the jump in physical properties across the interface leads to poor computational results. Therefore, the fluid properties near an interface are defined using a smoothed Heaviside kernel function, H_ϵ, given by:

$$H_\epsilon(\phi) = \begin{cases} 0 & , \phi < -\epsilon \\ \frac{1}{2}\left[1 + \frac{\phi}{\epsilon} + \frac{1}{\pi} sin\left(\frac{\pi\phi}{\epsilon}\right)\right] & , |\phi| < \epsilon \\ 1 & , \phi > \epsilon \end{cases} . \tag{2.31}$$

The fluid properties are thus defined as:

$$\rho(\phi) = \rho_1 H_\epsilon(\phi) + \rho_2(1 - H_\epsilon(\phi)), \tag{2.32}$$

$$\mu(\phi) = \mu_1 H_\epsilon(\phi) + \mu_2(1 - H_\epsilon(\phi)). \tag{2.33}$$

This results in a blending of the fluid properties near the interface over a region that is 2ϵ thick. Schematically, the interface and the region of property smoothing around it are shown in Fig. 2.1.

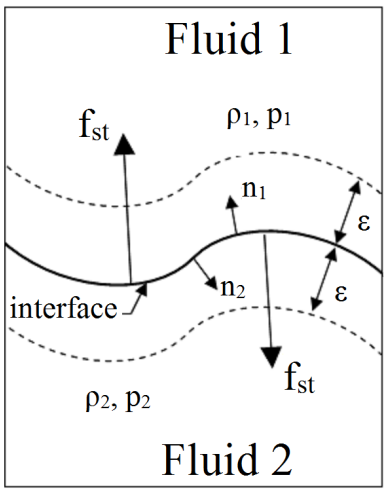


Figure 2.1: The interface and the region around it in which fluid properties are blended through the smooth Heaviside function. Image created by J. Rodriguez [65] and used with permission.

2.2.2 Redistancing: The Second Scalar

An exact distance field, d satisfies

$$|\nabla d| = 1 \tag{2.34}$$

where the gradient is positive in one phase and negative in the other. As described in Section 2.2.1, the material derivative (Eq. (2.30)) will transport the interface exactly, up to the accuracy of the discretization. However, for non-constant velocity fields, an initially perfect distance field will become distored due to the velocity gradients. This distortion occurs even under exact transport. Because material properties are blended across the interface, it is crucial to control this distortion, and this introduces the requirement for redistancing, or restoring the distance field to Eq. (2.34). Thus the level set is corrected by solving the following partial differential equation:

$$\frac{\partial d}{\partial \tau} = S(\phi)\Big[1 - |\nabla d|\Big], \tag{2.35}$$

where d is a scalar that represents the corrected distance field and τ is the pseudo time over which the equation is solved to steady-state. This may be more clearly expressed as the

following transport equation for d:

$$\frac{\partial d}{\partial \tau} + \vec{w} \cdot \nabla d = S(\phi)) \tag{2.36}$$

whose steady-state solution satisfies Eq. (2.34).

The pseudo velocity $\vec{w}$ that advects d is given as:

$$\vec{w} = S(\phi)\frac{\nabla d}{|\nabla d|}, \tag{2.37}$$

and $S(\phi)$ is defined as:

$$S(\phi) = \begin{cases} -1 & , \phi < -\epsilon \\ \left[\frac{\phi}{\epsilon} + \frac{1}{\pi} sin\left(\frac{\pi\phi}{\epsilon}\right)\right] & , |\phi| < \epsilon \\ 1 & , \phi > \epsilon \end{cases} . \tag{2.38}$$

Note that for discrete solutions of the level set, there are two sources of errors for the position of the interface. The first is in the advection of ϕ, the level set itself. The second, and separate, source is in the discrete solution of Eq. (2.36), which causes the interface to move while the distance field is being restored. Chapter 4 examines three redistancing variations with a particular focus on how they affect this second source of interface error.

An alternative to redistancing (that is not implemented in PHASTA) is given by Sethian [72, 73, 74]. In that approach, advancement of the level set is recast as a boundary value problem (rather than the initial value problem of redistancing) that advances the level set in such a way that an exact distance field is maintained, even in the presence of fluid velocity gradients, thereby eliminating the need for redistancing. This recasting of the level set allows for the use of computationally efficient fast marching methods (e.g., fast marching cubes [73]), and has given rise to widely used variational level set methods (see, e.g., [48, 47]) that are especially popular in image processing. Developing appropriate functions to advance the level set in complex flows (i.e., flows with merging and dividing interfaces), however, is a challenge that has limited this method's use in fluid dynamic simulations.

2.2.3 Modeling Surface Tension

For many two-phase fluid flow problems, interfacial motion induced by surface tension plays a significant role. One of the advantages of the level set is that it makes it relatively straightforward to model surface tension following the continuum approach proposed by Brackbill et al. [9] in order to represent the surface tension force as a body force (i.e., as a continuous three dimensional effect across an interface). The resulting body force due to surface tension can be written as

$$f = -\frac{\kappa(\phi)\nabla H_\epsilon(\phi)}{W}, \tag{2.39}$$

where $\kappa(\phi)$ is the curvature defined as

$$\kappa(\phi) = \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right), \tag{2.40}$$

and W is the Weber number given by:

$$W = \frac{\rho_1 L U^2}{\sigma}. \tag{2.41}$$

The surface tension term, Eq. 2.39 and local inter-facial curvature, Eq. 2.40 are easily represented in terms of the level set function. Because the level set function is a signed distance from the interface, the curvature of the interface can be accurately computed from the level set function. However, the calculation of curvature involves second order derivatives. Since PHASTA uses piecewise-linear basis functions, these second derivatives are zero on the element interior. Hence, the piecewise-constant gradients are reconstructed to be continuous using L_2 projection, and then the second order derivatives are evaluated by differentiating the reconstructed gradient. This procedure is described in detail by Jansen et al. [33]. The particular form of the surface tension force as a body force used here is from Chang et al. [14].

2.2.4 Finite Element Formulation

Equations (2.30) and (2.35) for the first and second scalars, respectively, can be represented by a single scalar advection equation of the form

$$\theta_{,t} + a_i \theta_{,i} = S \tag{2.42}$$

In case of the level set equation, $\theta = \phi$, $\boldsymbol{a} = \boldsymbol{u}$ and the forcing function S is zero. On the other hand, in the case of the redistancing equation, $\theta = d$, $\boldsymbol{a} = \boldsymbol{w}$, and S is given by Eq. (2.38). This section presents the general finite element formulation for the scalar advection equation, Eq. (2.42). The solution strategy for solving this equation is similar to that used for the equations of motion, as discussed in Section 2.1.2.

In keeping with the approach of the finite element method described in Section 2.1, Eq. (2.42) is cast in the residual form:

$$\theta_{,t} + a_i \theta_{,i} - S = 0. \tag{2.43}$$

To derive the weak form, Eq. (2.43) is multiplied by a smooth weighting function ψ belonging to a space of functions $\psi \in \Psi_h$ (the scalar counterpart of the vector space defined in Section 2.1.2). The product is then integrated over the spatial domain. Because the equation is solved in the convective form, there is no integration by parts, and therefore there are no boundary integrals. The resulting weak form is:

Find $\theta \in \Theta_h$ (the scalar solution space) such that,

$$\int_\Omega \left(\psi\theta_{,t} + \psi a_i \theta_{,i} - \psi S\right) d\Omega + \sum_{e=1}^{n_{el}} \int_{\bar{\Omega}_e} \{\hat{\mathcal{L}}^T \psi\tau \left(\theta_{,t} + a_i\theta_{,i} - S\right)\} dx = 0 \tag{2.44}$$

for all $\psi \in \Psi_h$. $\hat{\mathcal{L}}^T$ is the advective operator $u_i \frac{\partial}{\partial x_i}$. Here τ is the stabilization parameter defined as:

$$\tau = \frac{C}{\sqrt{c_1/\Delta t^2 + c_2 u_i g_{ij} u_j}}, \tag{2.45}$$

where C, c_1, and c_2 are defined based on the one-dimensional, linear advection-diffusion equation using a linear finite element basis, and $g_{ij} = \xi_{k,i}\xi_{k,j}$ is the covariant metric tensor

related to the mapping from global to element coordinates. Next, the weight functions (ψ), the solution variable (θ), and its time derivative ($\theta_{,t}$) are expanded in terms of linear basis functions. The integrals are then evaluated using Gaussian quadrature resulting in a system of ordinary differential equations which can be written as:

$$\boldsymbol{M}\dot{\theta} = \boldsymbol{N}(\theta). \tag{2.46}$$

The system of ordinary differential equations (linear when $\theta = \phi$, and non-linear when $\theta = d$) is converted to an algebraic system by introducing a time integrator (which is usually generalized-α for $\theta = \phi$, and backward Euler for $\theta = d$, although other choices are possible).

2.2.5 Volume Constraint on Redistancing

Although in a continuous solution of the redistancing equation, Eq. (2.35), the interface does not move, once the equation has been discretized, its solution tends to result in some movement of the interface. To minimize the motion of the interface, an additional constraint is imposed to preserve the original volume of each phase during the redistancing step. This constraint was proposed by Sussman et al. [78] and Sussman and Fatemi [79]. The distance field resulting from each pseudo time step during redistancing is constrained to minimize the total change in volume from the previous iteration. Although the volume constraint is already implemented within PHASTA, more complete discussion of the volume constraint is presented in Chapter 4, with the other level set variations.

Chapter 3

Coupled Level Set Volume of Fluid Development

In CLSVOF methods, the volume of fluid and level set fields are coupled together to provide better interface tracking than either method provides alone. This is typically done by solving the fluid motion, using the updated fluid motion to advect the level set (ϕ) and volume fraction (F) fields, and then using the information from both ϕ and F to update the interface and correct the level set field. This work follows that approach, so the general algorithm is given by Algorithm 3.1. PHASTA already contains a flow solver and means of advecting the level set, so the focus of this work is implementing a means of advecting a volume fraction field, reconstructing the interface from both volume fraction and level set information, and a means of updating the level set field away from the interface.

```
Solve the fluid equations
Update the fluid solution, u
Advect the level set using the updated fluid solution (u, φ → φ)
Advect the volume fraction using the updated fluid solution (u, F → F)
Reconstruct the interface from F and φ
Update φ in the rest of the domain (away from the interface)
```

Algorithm 3.1: CLSVOF algorithm

3.1 Volume of Fluid

Prior to this work, PHASTA had no implementation of a VOF method or other cell-based advection scheme. This work did not develop a stand-alone VOF scheme for PHASTA; it created the cell-based and face-based structures for advecting the volume fraction field, but did not implement an interface reconstruction scheme that would work solely from the volume fraction field. Because interface reconstruction in a CLSVOF scheme relies on information from the level set, development of a stand-alone VOF reconstruction scheme was considered unnecessary.

This work, instead, implements in PHASTA a simple volume fraction advection scheme. The volume fraction field F is a cell-centered scalar field that indicates the portion of the cell that is occupied by one fluid (usually the liquid phase). F takes on a value of 1 in cells that are occupied solely by the first fluid, 0 in cells occupied by the second fluid, and fractional values in cells containing the interface. The volume fraction is advected with the flow:

$$\frac{\partial F}{\partial t} + \tilde{u}_i \frac{\partial F}{\partial x_i} = 0, \tag{3.1}$$

where $\tilde{u}_i = u_i + \overset{\Delta}{u}_i$ and $\overset{\Delta}{u}_i$ is the conservation velocity from Eqs. (2.10) and (2.12).

3.1.1 Spatial and Temporal Discretization

The spatial discretization of the volume fraction advection problem is carried out in a manner similar to that of the incompressible Navier-Stokes equations and scalar advection equation, as discussed in Sections 2.1.2 and 2.2.4, respectively. However, there are some differences, since F is a cell-centered field rather than a nodal field and is piecewise constant rather than continuous. Choosing a weighting function, φ, that is also piecewise continuous leads to a first-order discontinuous Galerkin formulation for F instead of the continuous Galerkin formations seen above.

Taking the piecewise constant weighting function w, multiplying the volume fraction advection equation by it and integrating the product over the spatial domain yields

$$\int_{\Omega} wF_{,t}d\Omega + \int_{\Gamma} w\tilde{u}_i F n_i d\Gamma = 0, \tag{3.2}$$

where n_i are the unit normal (outward pointing) vectors on the boundary Γ. For each element, the resulting equation is

$$\int_{\Omega_e} wF_{,t}d\Omega_e + \sum_{f=1}^{n_f} \int_{\Gamma_f} w_f \tilde{u}_{n_f} F_f d\Gamma_f = 0, \tag{3.3}$$

where the f subscripts indicate the element's faces, and $\tilde{u}_{n_F}$ is the dot product of $\tilde{u}_i$ and n_i. Carrying out the spatial integrals (with one point quadrature at the center of each face) produces

$$F_{,t} = -\sum_{f=1}^{n_f} \tilde{u}_{n_f} F_f A_f, \tag{3.4}$$

where F_f is the volume fraction of the upwind element and A_f is the area of the face, f. Eq. (3.4) is simply a statement that the change in the volume fraction of an element is equal to the sum of the volume fraction fluxes across the element's faces.

Eq. (3.4) is then marched in time using an explicit Euler method:

$$F_e^{(n+1)} = F_e^{(n)} - \Delta t \sum_{f=1}^{n_f} \tilde{u}_{n_f} F_f A_f, \tag{3.5}$$

where the superscripts denote the values of the current and previous time steps and Δt is the time step itself. This simple upwinding has the obvious disadvantage of being sensitive to the time step size, but it also has the advantages of being relatively computationally fast (especially if A_f is computed and stored as a pre-processing step) and straightforward to implement. Higher order variants are possible by allowing F within an element to vary as a discontinuous function higher than piecewise constant but those higher order discontinous Galerkin methods were not pursued in this work.

3.1.2 Implementation in PHASTA

Implementing the volume fraction solution, Eq. (3.5), in PHASTA required several changes to accommodate the element-based method. For serial cases, these changes were straightforward. The simplest and most obvious of these was the addition of a data structure to store F, the size of which is equal to the number of elements. The second change needed was a data structure to keep track of each element's neighbors, called *ieneigh*. *Ieneigh* is of size number of elements by number of faces per element (which is four in the case of an all-tetrahedral mesh, as all of the meshes used here are). For a given element, its four entries in *ieneigh* are the global element numbers of its neighbors across each face. The definition of neighbors includes elements that match across a periodic boundary; if an element face is on a non-periodic problem boundary, the corresponding entry in *ieneigh* is zero, signifying there is no neighbor for that face. *Ieneigh* is generated as part of the mesh pre-processing. The third data structure added is one to store the normal vector for each face of each element. Because the normal vectors are used in every computation of Eq. (3.5), it is computationally more efficient to compute them in a preprocessing step and store them rather than to compute them each time they are needed.

For parallel cases, the changes are somewhat more complex. To make parallel computation of Eq. (3.5) efficient, ghost elements are used to allow the computation to be carried out on-processor after a single communication. To support this ghosting, the data structures for F and the element face normals are expanded so that on each processor, their size is the number of elements plus the number of ghost elements. The number of ghosts is determined by the mesh partitioner, which also generates the data structure needed for the face-based communication that populates the data for the ghost elements. As generated by the partitioner, the data structure matches the correct elements across part boundaries, but if an element has multiple faces on a part boundary, its corresponding ghosts may not be in face order. To account for that, there is an extra pre-processing step that checks face areas and

normal directions to ensure that the correct ghosts are matched with the corresponding faces in *ieneigh*.

3.1.3 Verification of Correctness

Implementing any new algorithm or method comes with its share of challenges in verifying that the algorithm is implemented correctly. In implementing this VOF method in PHASTA, there were particular challenges in ensuring that the new data structures behaved and communicated properly (e.g., ensuring that each element's neighbors corresponded to correct face of the element). To that end, a few strategies were particularly helpful. The first was to implement a uniform volume fraction field (either 0 or 1 everywhere) and check that an advection step still yielded a uniform field. The second was to implement a varying volume fraction field (e.g., the plug flow discussed in Sec. 3.1.5) but with zero velocity and check that the volume fraction field remained unchanged. The third strategy was to check that the total change in mass indicated by the volume fraction field over the whole domain after each advection step (in a case with non-zero velocity) was zero. This was easily done through *write* statements that had the added benefit of showing that the volume fraction advection had been solved, and because it did not require changing any of the problem set-up, could be used on any problem with no additional effort.

3.1.4 Additional Considerations

The volume fraction is not entirely independent of the level set. As a preprocessing step, the volume fraction field is initialized from the level set field, which is set in the modeling software as part of the problem setup. This initialization of the volume fraction field from the level set occurs at the beginning of the problem (before the first time step). The volume fraction can be initialized in two ways. The first option is to integrate the smoothed Heaviside function (Eq. 2.31) over each element. The second option is to geometrically determine the portion of each cell cut by the interface that is occupied by the first phase. This work used

the latter option because it is consistent with the work done to reconstruct the interface in the CLSVOF algorithm (see Sec. 3.2.2).

3.1.5 VOF Performance

Prior to being coupled to the existing level set algorithms, the VOF algorithm was tested on its own on a few simple problems. Because this is not a complete VOF algorithm (i.e., it does not include interface reconstruction), it was tested only on "red water/blue water" problems (meaning there are no differences in fluid properties between the two phases) with applied velocities, where interface dynamics do not contribute to the fluid solution. The algorithm was tested in three different cases: plug flow in a cubic domain, a rotating semi-circle, and an advecting sphere.

3.1.5.1 Plug Flow

The simplest of the test problems is plug flow through a cube. The problem domain, as shown in Fig. 3.1, is a simple cube that is 0.1 m on each edge with flow of 1 m/s along the x-axis, periodic boundaries on the cube walls perpendicular to the flow, and slip-walls on the other walls. The problem is a red water/blue water problem that is initialized with half the domain, in a slice perpendicular to the flow direction, occupied by one phase. Velocity is applied, so there is no flow solution. Additionally, there are no differences in material properties between the two phases, and there is no surface tension. The planar geometry of the interface and the lack of complicated flow physics make this problem a basic test of an algorithm's ability to handle a moving interface. This problem also provides a good test of whether periodicity has been properly implemented.

The problem was run for one domain transport time on two different mesh sizes. The coarse mesh has an element size of 0.01 m (or 10 elements per model edge) for a total of approximately 7k elements. The fine mesh has an element size of 0.005 m (or 20 elements per model edge) for a total of approximately 50k elements.

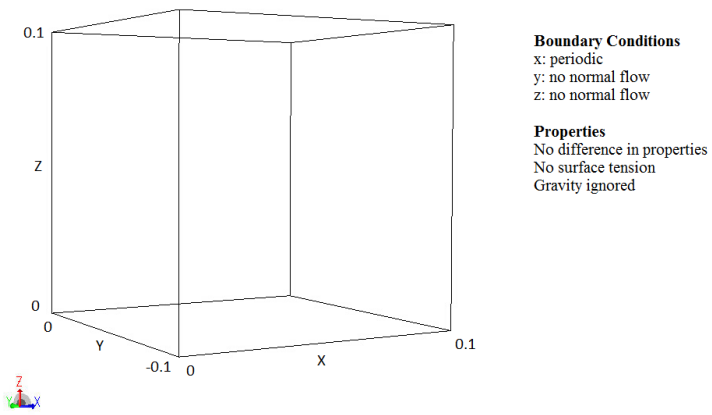


Figure 3.1: The problem domain for plug flow through a cube.

Evaluating the performance of the VOF algorithm on the plug flow problem showed that although the interface (as determined by where $F = 0.5$) was maintained over the course of a single domain transport time, the volume fraction field diffused numerically throughout the problem domain, and the numerical diffusion directly correlated with the element size. Fig. 3.2 shows for both meshes the initial and final conditions for the volume fraction field on a cut through the domain parallel to the flow direction (flow is to the right).

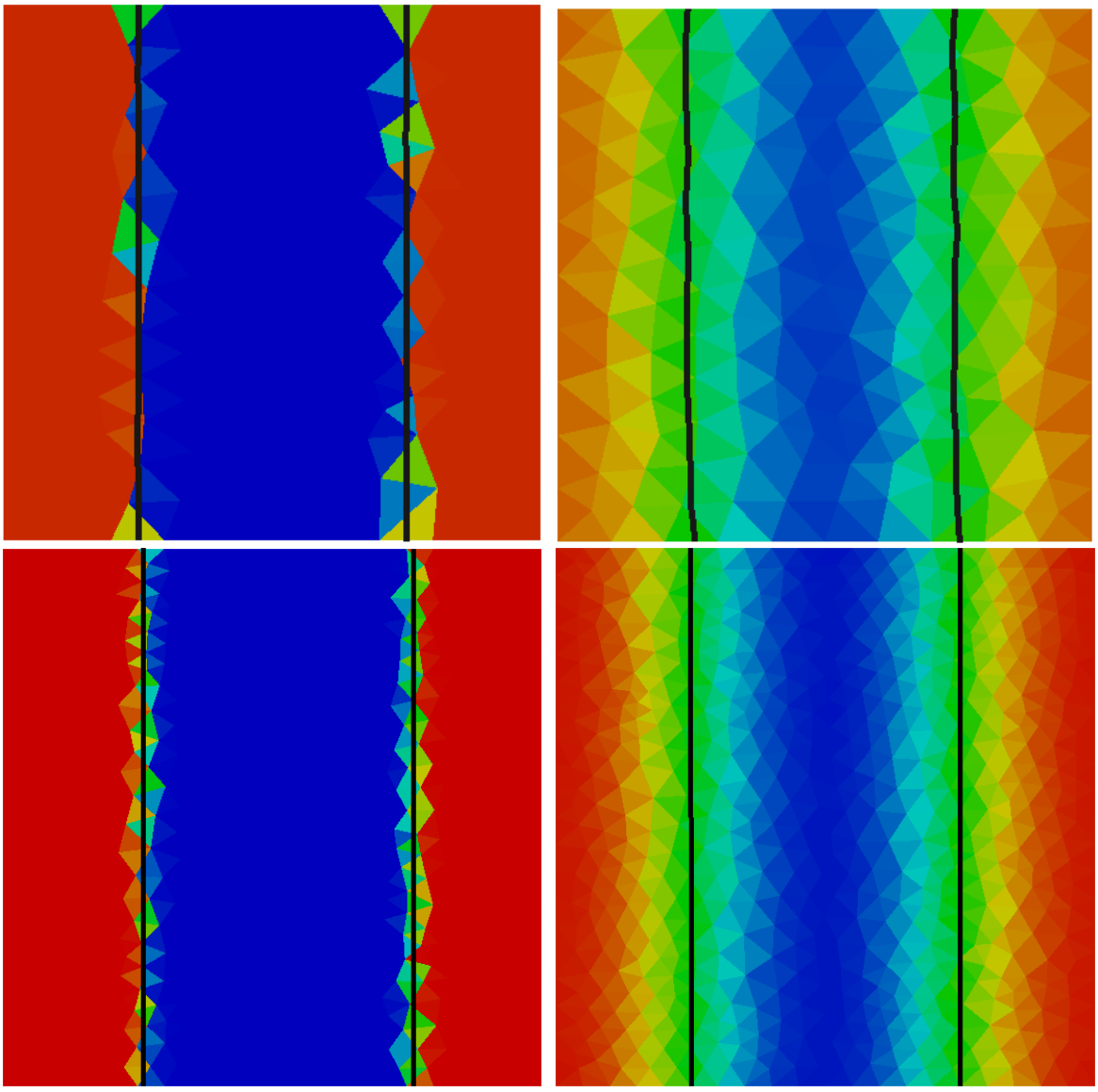


Figure 3.2: Initial condition of the volume fraction field in the cube (left) and after one domain transport time (right) for the coarse (top) and fine (bottom) meshes. The black lines indicate where $F = 0.5$, which is the interface inferred from the volume fraction field.

3.1.5.2 Rotating Semi-Circle

The second test problem for the VOF algorithm is a rotating semi-circle. The problem domain, as shown in Fig. 3.3, consists of a semi-circle of radius $R = 0.26$ m placed in the center of the 1 m x 1 m domain, which has periodic boundaries in each direction. The domain has a purely rotational applied velocity, $\omega = 2.0$ rad/s. This is a red water/blue

water problem, with no differences in properties between the two fluids, and there is no surface tension. This problem evaluates the VOF algorithm's behavior for more complex interfaces, including sharp corners.

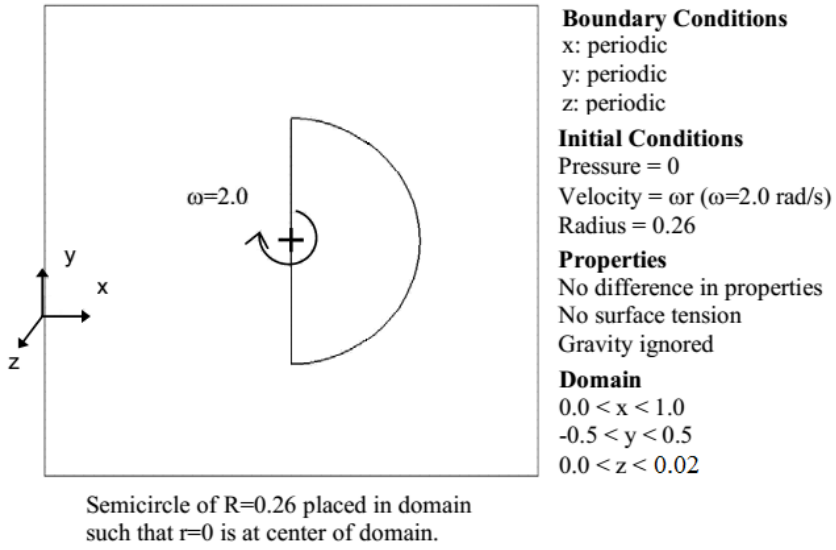


Figure 3.3: The problem domain for a rotating semi-circle. Image modified from one created by J. Rodriguez [65]; used with permission.

The problem was run for one complete rotation on two different mesh sizes. The coarse mesh has an element size of 0.05 m (or 10 elements per semi-circle diameter) for a total of approximately 3k elements. The fine mesh has an element size of 0.025 m (or 20 elements per semi-circle diameter) for a total of approximately 16k elements.

Fig. 3.4 shows the initial and final (after one complete rotation) condition of the semi-circle. In this case, there are two things to note. The first is that in the initial condition (the left side of Fig. 3.4), the interface determined by $F = 0.5$ does not quite maintain the sharp corners (see Fig. 3.3 for comparison) of the semi-circle specified in the problem setup and by the level set field (from which the volume fraction field is initialized). The second thing to note is that the diffusion of the volume fraction field is stronger in the flow direction, resulting in the backwards "C" shape seen after a complete rotation. As with the plug flow problem, the extent of the numerical diffusion correlates with the element size.

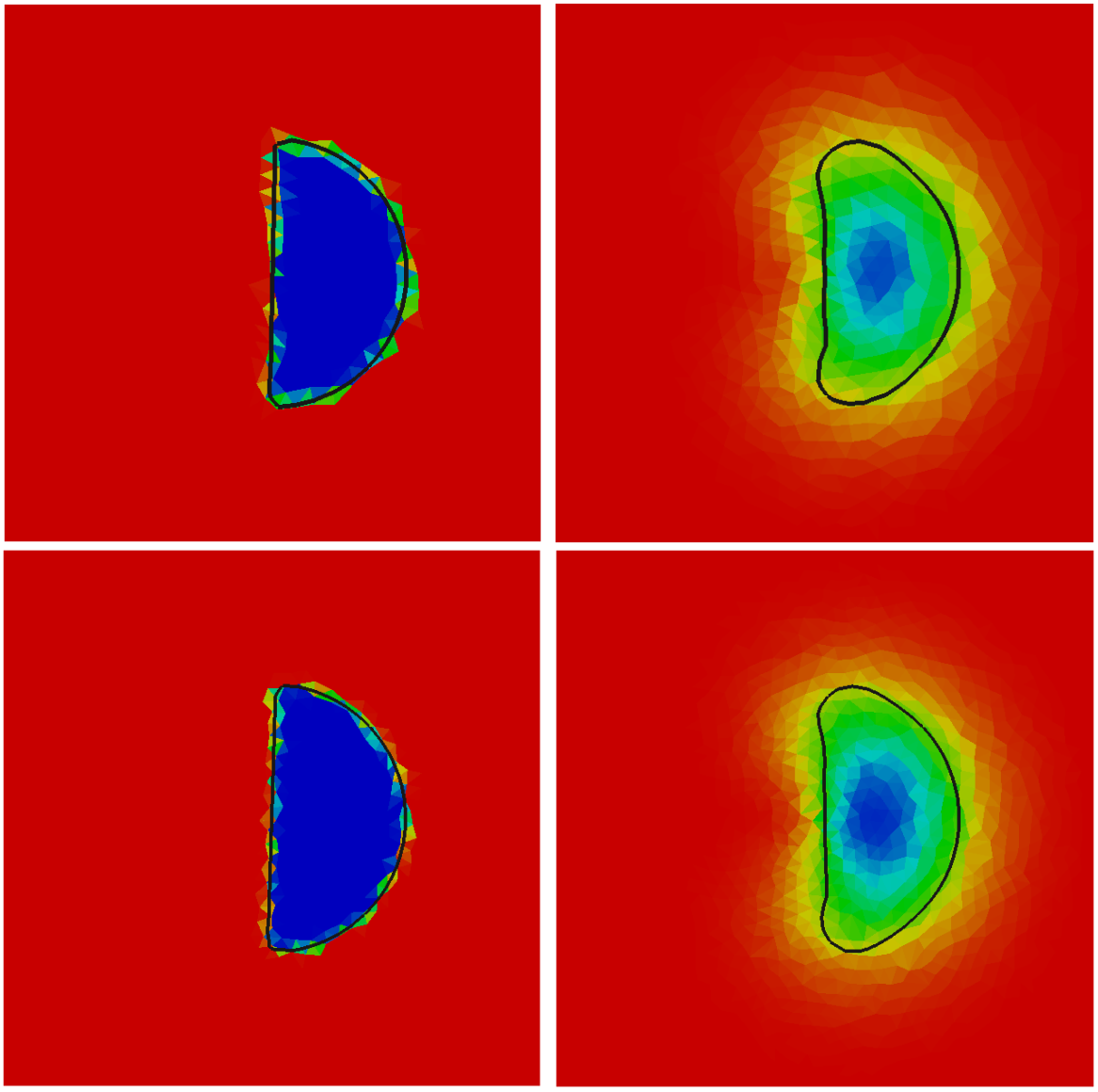


Figure 3.4: Initial condition of the volume fraction field for the rotating semi-circle (left) and after one complete rotation (right) for the coarse (top) and fine (bottom) meshes. The black lines indicate where $F = 0.5$, which is the interface inferred from the volume fraction field.

3.1.5.3 Advecting Sphere

The advecting sphere is a red water/blue water problem (there are no differences in properties between the two fluids) with an applied velocity field along the x-axis of the problem domain. The interface for the advecting sphere lacks the sharp corners of the

rotating semi-circle, but presents a curved three-dimensional interface that is more complex than the planar interface of the plug flow. As shown in Fig. 3.5, the problem is initialized with a sphere of one fluid centered in the physical domain while the second fluid occupies the rest of it. There is no surface tension. Like the plug flow, the domain for the advecting sphere has periodic boundaries in the x-direction and slip walls in the y- and z-directions. The advecting sphere is a useful problem for evaluating the CLSVOF algorithm's ability to reconstruct curved interfaces and for further investigating the effect of the volume fraction's numerical diffusion on the interface reconstruction.

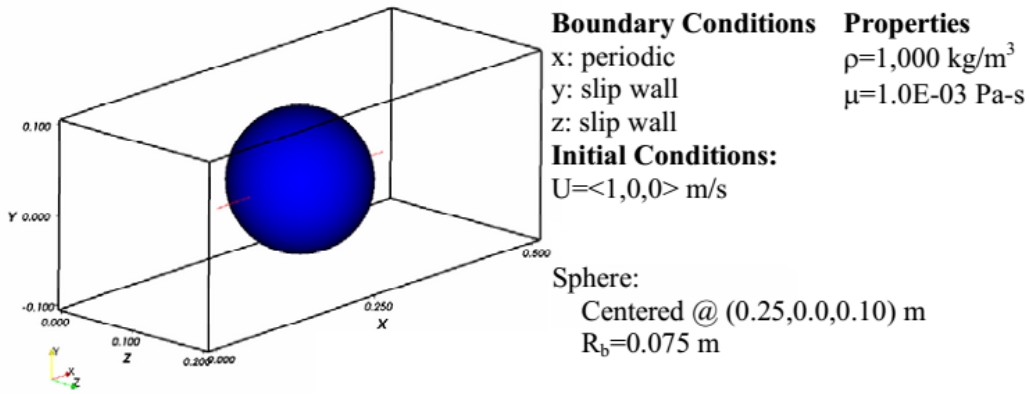


Figure 3.5: The problem domain for an advecting sphere. Image modified from one created by J. Rodriguez [65]; used with permission.

The problem was run for one domain transport time on two different mesh sizes. The coarse mesh has an element size of 0.025 m (or 6 elements per circle diameter) for a total of approximately 9k elements. The fine mesh has an element size of 0.0125 m (or 12 elements per circle diameter) for a total of approximately 66k elements.

Fig. 3.6 shows the initial condition and the final condition after one domain transport time for a cut plane through the center of the advecting sphere on the coarse mesh. Fig. 3.7 shows the same conditions for the advecting sphere on the fine mesh. These results show, even more strongly than for the rotating semi-circle, that the diffusion of the volume fraction field is preferentially in the flow direction and that the diffusion directly correlates with mesh size.

They also show, that despite the numerical diffusion, the interface (as defined by $F = 0.5$) is well maintained.

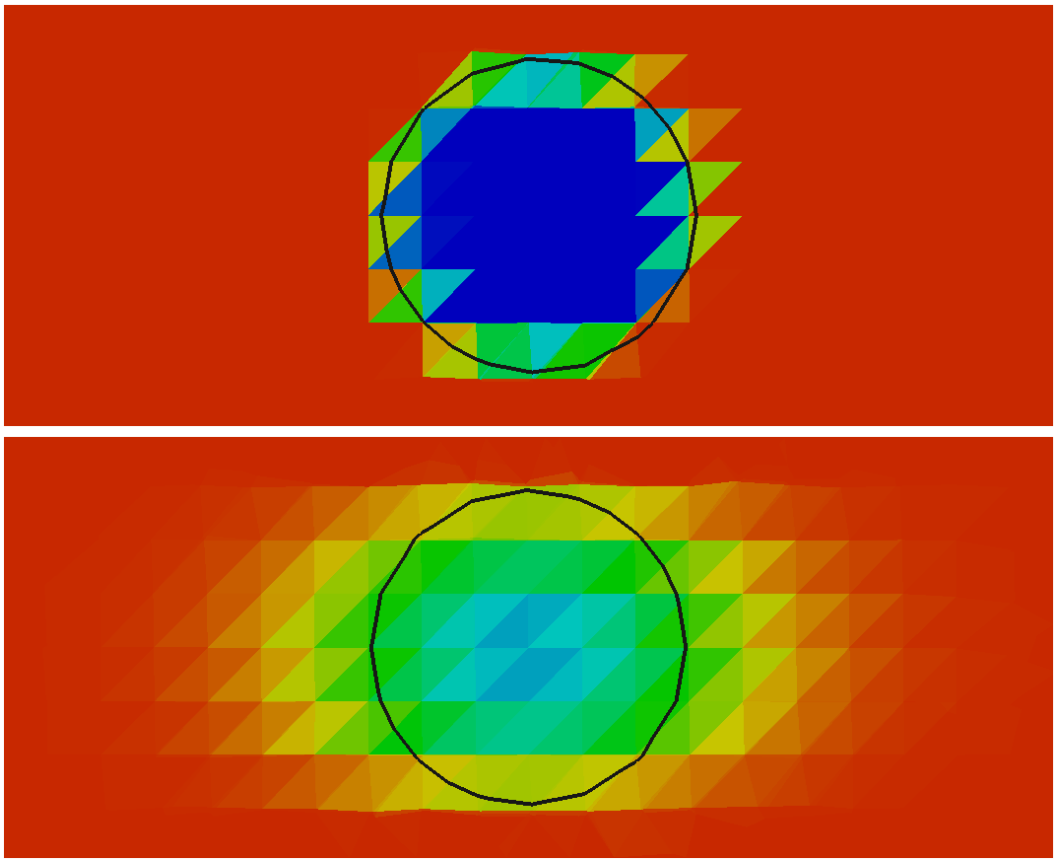


Figure 3.6: Initial condition of the volume fraction field in the sphere (top) and after one domain transport time (bottom) for the coarse mesh. The black lines indicate where $F = 0.5$, which is the interface inferred from the volume fraction field.

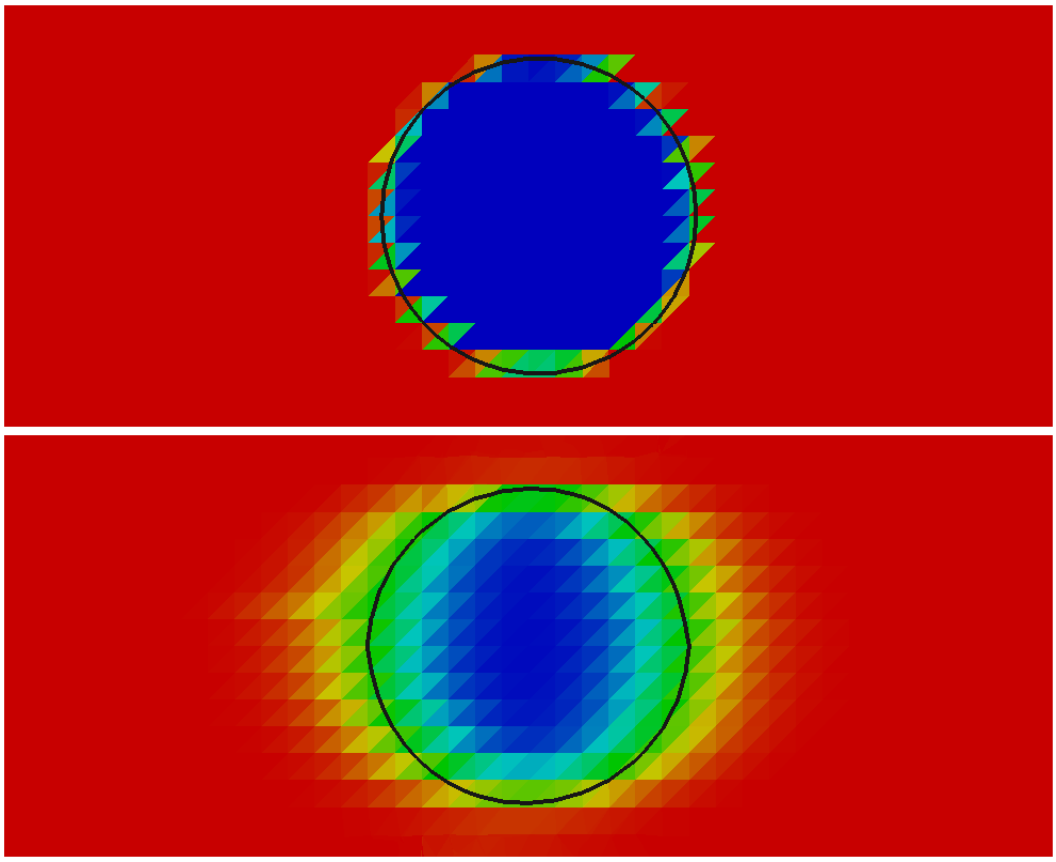


Figure 3.7: Initial condition of the volume fraction field in the sphere (top) and after one domain transport time (bottom) for the fine mesh. The black lines indicate where $F = 0.5$, which is the interface inferred from the volume fraction field.

3.1.5.4 VOF Summary

As seen in these three test problems, the advection of the volume fraction is quite numerically diffusive; without some sharpening mechanism, the volume fraction of each cell tends toward the volume fraction of the entire domain. This diffusivity can be countered somewhat by the use of more refined meshes, increasing the accuracy of the volume fraction advection. The current implementation is first-order accurate in both space and time; a higher order method would also improve the behavior of the volume fraction.

Because the time integration of the volume fraction advection equation is explicit, it is quite sensitive to time step size. While in theory, the CFL number only needs to be less

than one, trial and error has shown that for stability the CFL number (based on the average element edge-length) for this scheme on an unstructured mesh actually needs to be no larger than ~ 0.2. This is because the highest CFL number in the mesh is significantly larger than the one based on average edge-length, due to the nature of the unstructured tetrahedral meshes used here.

3.2 Coupling the Level Set and Volume of Fluid

3.2.1 Planned CLSVOF Algorithm

The novel aspect of CLSVOF methods lies in the reconstruction of the interface and nearby level set field. The algorithm used in this work is based on the level set volume of fluid coupling algorithm of Sussman and Puckett [81], which is presented in simplified form in Algorithm 3.2.

There are a few things to note about the Sussman and Puckett algorithm. The first is that the advection scheme allows for non-physical values of the volume fraction, which means that the volume fraction must be truncated in cells where it is less than zero or greater than one. This truncation also means that while their CLSVOF method offers improved volume conservation over level set methods, it is not strictly volume conserving. The second thing to note is that both the level set and volume fraction fields are cell-centered fields, which means that when the algorithm refers to ϕ changing signs, it means between the current cell and any neighboring cell. Third, the Sussman and Puckett algorithm is for a structured grid, which makes it straightforward to both know all of a cell's neighbors and to compute the distance between any given cell and any other cell. Finally, the procedure of updating level set values within some distance of the interface is computationally repetitive (most cells get checked many times) and poorly suited to parallelization (because each cell containing a piece of the interface has to communicate with every other cell within the distance for which the level set is being recomputed).

(1) Truncate the volume fraction $0 \leq F \leq 1$

(2) Tag all cells

  (a) In each cell, check whether ϕ changes signs

    (i) If F is fractional and ϕ changes sign, linearly construct the interface plane

    (ii) For each cell within some distance

      (ii.a) Compute the distance from that cell to the current cell's ϕ

      (ii.b) Update that cell's ϕ

      (ii.c) If that cell is tagged, ϕ is the computed distance with the correct sign

      (ii.d) If that cell is untagged, ϕ is the minimum magnitude of the current value and the computed distance with the correct sign

      (ii.e) Untag the cell

  (b) Untag the interface cell

(3) For all cells still tagged, set ϕ to an arbitrarily large magnitude, keeping the correct sign

Algorithm 3.2: Sussman and Puckett's coupling of the level set and volume fraction fields

Implementing a similar algorithm in PHASTA, where the level set is a nodal (rather than cell-centered) field, and where the objective is to support parallel computation on unstructured meshes, requires at least two modifications. First, the new algorithm must ac-

commodate both cell-centered and nodal fields. Second, the procedure for updating the level set away from the interface must be modified so that each node needs to communicate only with the other nodes with which it is edge-adjacent. The initial algorithm that incorporates these modifications is given in Algorithm 3.3.

(1) Tag all nodes

(2) In a loop over all elements, do the following:

  (a) Find all cells containing the interface ($0 < F < 1$ and ϕ changes signs across the cell)

  (b) For each cell containing the interface

    (i) Reconstruct the interface using F and ϕ

    (ii) Reconstruct the level set at each of that cell's nodes

    (iii) If ϕ at the node has already been reconstructed, the value that is kept is the sign and minimum magnitude of the already reconstructed value (from another cell) and the reconstructed value for the current cell.

  (c) For all elements that did not contain the interface, reset the volume fraction to 0 or 1, as indicated by the sign of the level set

(3) Update ϕ away from the interface by "walking out" from the interface elements until ϕ reaches a set magnitude

(4) Set ϕ in the rest of the domain

Algorithm 3.3: Planned coupling of the level set and volume fraction fields for implementation in PHASTA

Algorithm 3.3 is a high-level algorithm that served only as a starting point for CLSVOF implementation. Details for the interface reconstruction and updating of ϕ (both for nodes belonging to interface elements and nodes away from the interface) are given in the following sections, which also discuss further modifications that were found to be necessary.

3.2.2 Interface Reconstruction

The reconstruction of the interface from the volume fraction and level set fields is local to each element that contains a piece of the interface ($0 < F < 1$, and ϕ changes signs across the element). The interface is piecewise linear, so in each element, the level set determines the normal of the interface and the volume fraction determines where in the element it falls. The planned reconstruction algorithm is given by Algorithm 3.4.

There are two things to note in Algorithm 3.4. The first is that in step 1 (finding the intersection points between the interface plane and the element's edges), there could be a total of four points of intersection between the interface plane and the tetrahedral element (all-tetrahedral meshes are assumed in this development), but since the interface is piecewise linear, the fourth intersection point (if it exists) is not used in defining the interface plane. The second is that Step 6b does not guarantee the correct value of ϕ at each node because it uses only the local interface information of the current element, and in some cases (e.g., the planar interface shown in Fig. 3.8), the correct distance to the interface cannot be found locally. In other cases (e.g., the convex side of a curved interface), the local distance to the interface facet is correct, even when that distance lies along an element edge. This is discussed further in Sec. 3.2.3.

(1) Loop over the edges of the element to find the points at which $\phi = 0$

(2) Use the first three intersection points found to define the interface plane

(3) Loop over the nodes of the element to find the maximum distances in each direction (positive ϕ and negative ϕ) that the interface can move without leaving the element

(4) Compute the relative error between the volume fraction given by the interface plane and the volume fraction given by F

(5) Use recursive bisection to slide the interface plane along its normal until the relative error in volume fractions is sufficiently small

(6) Loop over the nodes of the element to update their level set values based on the new interface plane

  (a) Project the node onto the interface plane

  (b) Check whether the projected point is within the element

    (i) If the projected point is within the element, then the new level set value is simply the distance between the interface plane and the node

    (ii) If the the projected point is outside the element, then the new level set value is the shortest distance from the region of the interface plane inside the element to the node (see Fig. 3.8)

Algorithm 3.4: Planned reconstruction of the interface from the volume fraction and level set fields

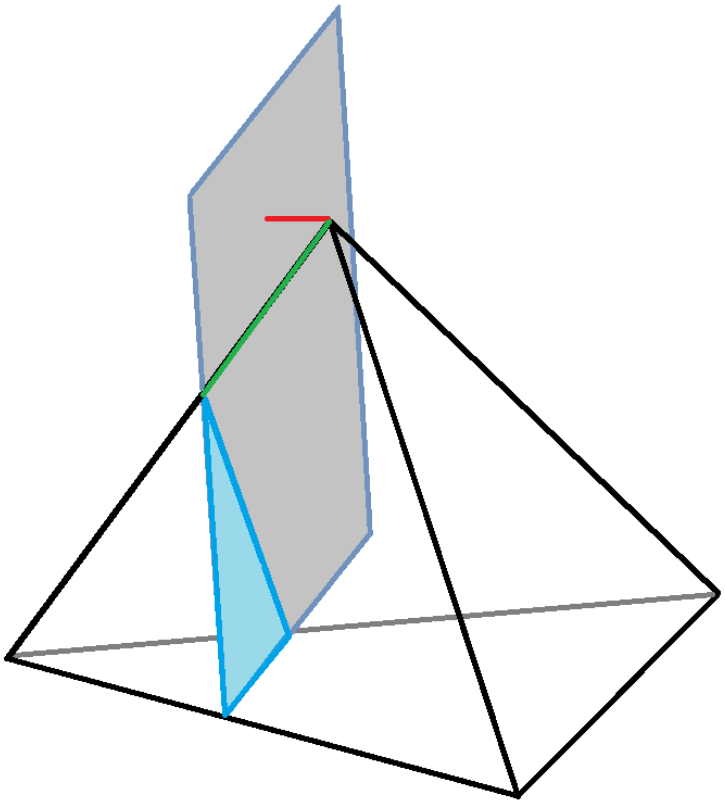


Figure 3.8: For this element, the distance from the uppermost node to the interface plane (shown in red) lies outside the element. The distance used for the level set (shown in green) is the shortest distance to the interface facet (shown in blue) rather than the interface plane (shown in gray). In cases where the shortest distance from a node to the interface plane lies outside an element, the shortest distance to the interface facet will always lie along an edge of the element.

3.2.2.1 Verification of Correctness

Because Algorithm 3.4 is a single-element algorithm, it was first implemented in MATLAB [51] for verification. This enabled easy evaluation of each step of the algorithm because nodal coordinates, values of ϕ, and volume fractions could be chosen to allow the algorithm's results to be readily checked against hand calculations. Once the algorithm was shown to

be correctly implemented in MATLAB, it was implemented in PHASTA where minimal additional debugging was required. This strategy of initial implementation in a simplified environment greatly accelerated the development process.

3.2.3 Updating ϕ for Nodes Belonging to Interface Elements

In Algorithm 3.3, updating the values of ϕ for nodes belonging to interface elements is a local operation, performed with only the information from the current interface element and a tag indicating whether that node had been previously updated. This turned out to be an infeasible approach for three reasons. First, as mentioned in Sec. 3.2.2, local element interface information is insufficient to compute a correct distance to the interface. In the case of a planar interface, as shown in Fig. 3.9, it is possible for the correct distance from a node to the interface to be on a path that is not contained by the interface element the node belongs to. However, in other cases, such as a node being on the convex side of a curving interface, simply accepting the distance to the element's interface plane results in too small a magnitude for the level set. To address this, the distance between every node belonging to an interface element and every interface facet must be calculated; then, the minimum magnitude result (keeping the original sign of ϕ) is the correct distance. The second reason the proposed local update scheme is infeasible is that except for the first interface element evaluated, interface reconstruction uses a mix of advected values of ϕ and locally reconstructed (possibly incorrect, as can occur with planar interfaces, for example) values of ϕ. This mixing of advected and reconstructed values of ϕ can be avoided by using only advected values to reconstruct the interface as a set of facets and then using the reconstructed interface facets from all of a node's adjacent elements to determine the reconstructed ϕ for that node. In this way, reconstructing interface facets is a local operation, but reconstructing ϕ is not. The third reason is that, as proposed, there is no continuity requirement on the interface facets, which can lead to level set values that are too small in

magnitude, meaning that the level set is no longer a distance field. This third problem can be addressed by requiring the interface facets to be continuous.

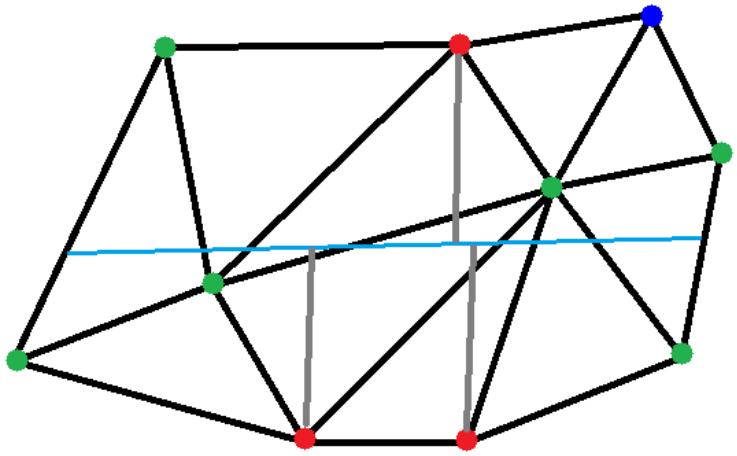


Figure 3.9: For the nodes in red, the correct distance to the interface (shown in light blue) lies along normals (shown in gray) that are not contained within the node's element.

There are two steps to enforcing facet continuity. First, motion of the interface facet is restricted so that it cannot change which edges of an element it intersects. This is particularly important since intersecting edge changes can lead to a node being assigned the wrong level set sign, creating a highly distorted interface, as defined by the level set. Second, all the intersection points on an edge are averaged. Neither of these steps is volume conserving, but both were necessary to reduce distortion of the interface.

These changes in how the interface nodes are handled primarily affect Step 6 of Algorithm 3.3. Instead, after the interface facet has been adjusted to meet the volume fraction (within the limitation that the facet cannot change edges of the element), the intersection points of the interface facet and the element edges are stored, and Step 6 of Algorithm 3.3 is replaced by Algorithm 3.5.

```
(1) Loop over the interface elements to push the facet data to
    the edges

(2) Loop over all the interface edges, averaging the intersection
    points on each edge

(3) Loop over the interface elements to compute the new facet
    normals

(4) Loop over the nodes belonging to interface elements

    (a) Loop over the interface facets, computing the distance to
        the node

    (b) Keep the minimum magnitude value and store as the updated
        level set

    (c) Mark the node as updated
```

Algorithm 3.5: Updating the level set value for nodes near the interface

A potential alternative to simply averaging the interface facet intersection points along an edge would be to find a volume-neutral edge value. This would require keeping track of all elements that are adjacent to an edge, so that as the interface intersection point on the edge is adjusted, the volume fraction for all of the adjacent elements can be kept constant (or at least have the change minimized). This could result in better volume conservation than the averaging that is implemented here. This possibility is discussed further in Chapter 6 in the section on future work.

3.2.3.1 Verification of Correctness

Verifying the process for establishing new level set values at the nodes of interface elements was more complicated than verifying the process of establishing the interface facets.

The most useful approach was to set an arbitrary volume fraction for all elements within a narrow region that also contained the interface and then, with no advection of the level set or the volume fraction, check both the interface facets and the reconstructed interface. Using a planar interface, such as that in the plug flow described in Sec. 3.1.5.1, provided a means to easily verify that the interface orientation from the level set was being preserved while the location within the elements corresponded to the prescribed volume fraction. Comparing the interface facets from the elements to the overall level set field also made it simple to check whether the minimization step (for nodes that belonged to multiple elements containing the interface) was being carried out properly.

3.2.4 Updating ϕ away from the Interface

In Sussman's CLSVOF method [81], the level set is reconstructed for all elements that are within a given distance of the interface. Sussman's use of regular rectangular grids makes this sort of reconstruction straightforward since it is easy to know which elements are within a given distance of an element containing a portion of the interface. The use of unstructured meshes makes Sussman's approach impractical, since the nodes within a set distance of the interface can change throughout the course of a simulation and the data structures to track those nodes would require frequent recomputation. Additionally, an element-based walkout procedure, in which the level set values of three nodes are used to compute the value for the fourth node, encounters difficulties when the nodes are aligned along Cartesian coordinates, as often happens on the interior of the unstructured meshes used here. While those difficulties could be overcome by coordinate transformations, there is no single transformation that would work for all elements. A walkout approach is therefore deemed infeasible for an unstructured mesh. Instead, a modified redistancing procedure is used to correct the level set away from the interface.

The modified redistancing procedure still solves Eq. 2.36, however, it does not include any volume constraint. Instead, all of the nodes that had updated level set values from

the interface reconstruction are treated like boundary nodes with an essential boundary condition; their level set value is not permitted to change. Because all of the level set values at nodes nearest the interface are set, the interface does not move at all during the solution of Eq. 2.36. In this way, the level set redistancing away from the interface is made to conserve volume.

3.2.5 CLSVOF Performance

As implemented here, the CLSVOF algorithm performs poorly, even on applied velocity problems where there is no velocity gradient to introduce distortions in the level set field. The reconstruction of the interface at each time step introduces small distortions that accumulate in a way that both produces a non-physical interface and fails to conserve volume. This poor behavior is particularly apparent in the plug flow problem (described in Sec. 3.1.5.1) and the advecting sphere problem (described in Sec. 3.1.5.3). For example, the left side of Fig. 3.10 shows the distortions that arise in the (initially planar) interfaces of the plug flow problem after a single time step, and the right side of Fig. 3.10 shows how the distortions accumulate over twenty time steps. Similarly, Fig. 3.11, shows the unacceptable effect of the distortions on the more complex three-dimensional interface of the advecting sphere problem. In both cases, the time step size corresponds to a CFL number of approximately 0.005, which is small enough that there are no instability effects from the volume fraction advection.

There are a number of possible contributing factors to this poor performance, including numerical instability of the volume fraction advection algorithm (resulting from too high a CFL number), numerical diffusion of the volume fraction field, and that the volume fraction and level set fields are computed at different places in the mesh. In these cases, numerical instability was removed from being a possible contributor by the choice of small time steps. The same time step was used for both the explicit volume fraction advection and the implicit level set advection, although it is not necessary to limit the time step for the implicit flow

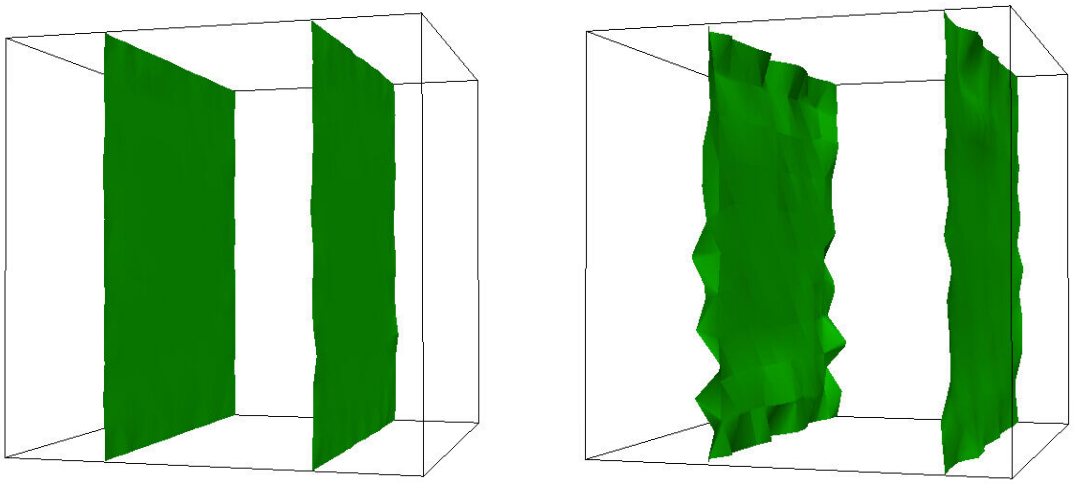


Figure 3.10: Interfaces for the plug flow problem after one time step (left) and after twenty time steps (right) resulting from the CLSVOF algorithm ($\delta t = 0.0005$, $CFL = 0.005$).

and level set solutions to that required for the volume fraction solution. Instead, the flow time step can be subdivided into several time steps for the volume fraction.

The numerical diffusion of the volume fraction, which can be reduced, though not eliminated, by the use of smaller time steps and finer meshes, is clearly a contributing factor to the distortions in the interface. Due to the numerical diffusion, some of the elements containing the interface end up with volume fractions that are closer to the domain volume fraction than they should be. Then, when the interface is reconstructed from the advected volume fraction and level set fields, the requirements for matching the volume fraction distort the level set's representation of the interface, which is the reconstructed interface. The distortion can be seen in Fig. 3.12, which shows the effect for a single node on the surface of the cube in the plug flow problem. In Fig. 3.12, the planar interface from the advected level set is in light gray, while the reconstructed interface facets for each element are in blue, and the final reconstructed interface is in green. The distortion in Fig. 3.12 is lessened but not eliminated when facet continuity is required.

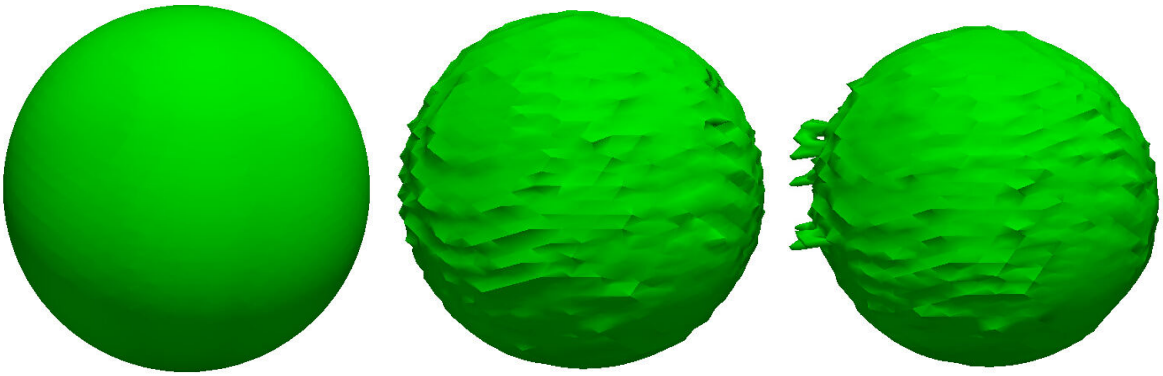


Figure 3.11: Interfaces for the advecting sphere problem initially (left), after 24 time steps (center) and after 48 time steps (right) resulting from the CLSVOF algorithm ($\delta t = 0.00025$, $CFL = 0.005$).

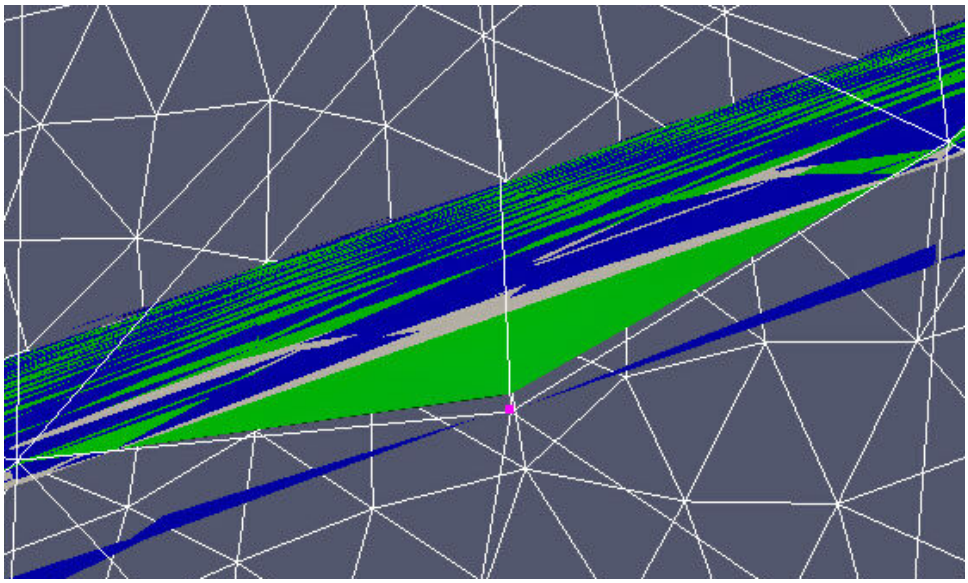


Figure 3.12: Distortion of a planar interface resulting from numerical diffusion of the volume fraction in the CLSVOF algorithm. Light gray shows the planar interface; blue shows the interface facets constructed from the volume fraction and level set fields, and green shows the reconstructed interface.

In addition to the distortion introduced by numerical diffusion of the volume fraction, the computation of new level set values (in reconstructing the interface) is itself a source of distortion. Because the level set field is a nodal field, the change to the level set value at a single node affects the level set field for every element containing that node. This means that

even though interface reconstruction was treated as an independent step for each element containing the interface, interface reconstruction for one element changes the interface in all adjacent elements containing the interface and further affects the level set field for all node-adjacent elements, regardless of whether they contain the interface. In the case of a planar interface where the volume fraction and level set agree on the interface location (for example, when the volume fraction is first established), this distortion is not apparent. However, for curved interfaces, even when the volume fraction and level set fields agree, the process of computing new level set values for the reconstructed interface changes the interface, in the direction of increasing the curvature. In Sussman and Puckett's work [81], the volume fraction and level set fields are colocated at the element centers, thereby avoiding the problems of elemental interdepence of the level set values.

3.2.6 Parallel Considerations for CLSVOF

Despite ultimately poor performance of the CLSVOF algorithm, one of the initial objectives was to develop an algorithm that would be compatible with parallel computing. Accordingly, consideration was given early in the algorithm's development to what would be necessary to support parallel computation. The conclusions of that effort are presented here to aid future development of a parallel CLSVOF algorithm.

For parallel computation, the steps of coupling the level set and volume fraction fields and reconstructing the interface can be done on the local processor without communication, except for the nodal update (Algorithm 3.5). For the nodal update process to be completed, any nodes on part boundaries need to know their values from other parts in order to compare them and find the minimum. So, in parallel cases, that step is allowed to remain incomplete for nodes on part boundaries until the CLSVOF algorithm has traversed every element on the part. Then, before updating the level set away from the interface (using the modified redistancing), all of the nodal communication takes place so that each node ends up with its global minimum rather than its on-processor minimum.

The communication phase to ensure that each node belonging to an element containing the interface gets its global minimum value has four stages. The first stage is a send/receive pair in which the remote copies send their updated average level set values to the owner copies. After the owner receives the values from the remote copies, the owner finds the average level set value for each node on the part boundary. That value is then communicated back to the remote copies via a send/receive pair that is the third stage of the communication. Finally, each part updates its values to be those that the owner determined were the correct ones.

Chapter 4

Level Set Variations and Results

In level set methods, the level set is a signed distance field advected with the flow. Velocity gradients distort the distance field; a redistancing procedure must be applied to restore the level set to a distance field. While various advection schemes have been used to solve the level set, the novelty in level set methods is in how the redistancing is handled. This work explores three possibilities for level set redistancing: simple redistancing without a volume constraint, redistancing with a global volume constraint, and redistancing that strictly prohibits motion of the interface. Each variation is examined with regards to its computational cost, accuracy, ease of implementation, parallelizability, volume conservation, and preservation of curvature.

4.1 Level Set Redistancing Variations

Section 2.2 describes how the level set is solved and redistanced. After the level set is solved, its value is assigned to a second scalar d, and redistancing is then conducted by solving the following equation:

$$\frac{\partial d}{\partial \tau} = S(\phi)\Big[1 - |\nabla d|\Big]. \tag{4.1}$$

The solution to Eq. 4.1 restores d to a distance field with a gradient equal to 1. After d is found, the level set takes on the value of d, and redistancing is complete.

4.1.1 Variation 1: No Redistancing Constraint

The simplest way to handle redistancing is to carry it out without placing any constraints on the redistancing solution. In this case, the small motions of the interface that result from discretization are both permitted and accepted without any modification. This offers the advantages of being easy to implement and not being computationally intensive, but it also has the significant disadvantage of potentially being quite non-conservative in volume. Within PHASTA, a lack of redistancing constraint is the base condition; the user must turn on one of the other options in order to use them.

4.1.2 Variation 2: Global Volume Constraint

The second redistancing variation, and the first to constrain the motion of the interface, is a global volume constraint based on the work of Sussman et al. [78] and Sussman and Fatemi [79]. The basis of the constraint is to require the volume of each phase to remain constant during each redistancing iteration. This constraint was implemented in PHASTA by Nagrath [54, 55, 56]. The constraint is applied in the following way.

The volume in each element can be defined as:

$$V^k = \int_{\Omega^e} H(d^k) d\Omega^e, \tag{4.2}$$

where H is the smoothed Heaviside function and d^k is the distance field at the k^{th} iteration of redistancing in pseudo time, τ^k. Over a psuedo-time step, the volume should not change; i.e., $V^k = V^0$. The difference in volume from one psuedo-time step to the next can be approximated by:

$$\begin{aligned} V^k - V^0 &\approx (\tau^k - \tau^0) \textstyle\int_{\Omega^e} \frac{dH'_\epsilon(d^0)}{d\tau} d\Omega^e \\ &\approx \textstyle\int_{\Omega^e} H'_\epsilon(d^0)(d^k - d^0) d\Omega^e, \end{aligned} \tag{4.3}$$

where

$$H'_\epsilon(d) = \left\{ \begin{array}{c} 0, |d| > \epsilon, \\ \\ \frac{1}{2}\left[\frac{1}{\epsilon} + \frac{1}{\epsilon} cos(\frac{\pi d}{\epsilon})\right], |d| \leq \epsilon. \end{array} \right\}. \tag{4.4}$$

Therefore, to minimize the change in volume, the current values of the distance function (the second scalar, which was initialized as the level set), denoted as $\tilde{d}^k$, are projected onto new values, denoted by d^k, which satisfy:

$$\int_{\Omega^e} H'_\epsilon(d^0)(d^k - d^0) d\Omega^e = 0. \tag{4.5}$$

The new distance field d^k has the form:

$$d^k = \tilde{d}^k + \lambda_{\Omega^e}(\tau^k - \tau^0) H'_\epsilon(d^0), \tag{4.6}$$

where λ_{Ω^e} is assumed to be constant in Ω^e, and is given by:

$$\lambda_{\Omega^e} = \frac{-\int_{\Omega^e} H'_\epsilon(d^0) \left(\frac{\tilde{d}^k - d^0}{\tau^k - \tau^0}\right) d\Omega^e}{\int_{\Omega^e} \left(H'_\epsilon(d^0)\right)^2 d\Omega^e}. \tag{4.7}$$

When the distance field is converged, $d^k = \tilde{d}^k$, and $\lambda_{\Omega^e} = 0$. The integrals for calculating λ_{Ω^e} in Eq. 4.7 are evaluated at the element level and are projected onto the global nodes by L_2 projection, as described by Jansen [33]. Then Eq. 4.6 is solved to obtain the constrained redistanced level set function. Within PHASTA, the volume constraint correction is applied globally after each redistancing iteration. The volume constraint as described here has been found to improve both the convergence and the volume conservation of the redistancing procedure; however, it does not require local volume conservation, and high curvature areas tend to be lost (e.g., small droplets disappear and their mass is redistributed among larger droplets).

4.1.3 Variation 3: No Interface Motion

An alternative to the global volume constraint is suggested by the approach for updating the level set at nodes away from the interface in the CLSVOF algorithm. In that

case, new level set values for the nodes belonging to elements containing the interface were computed from the advected level set and volume fraction fields, and then those nodes were treated as boundary nodes with essential boundary conditions while redistancing was applied to the rest of the domain. A similar approach can be employed in a level set only case to eliminate interface motion during redistancing. This can be achieved by identifying the nodes that bound the interface and treating them as boundary nodes with essential boundary condtions without changing their advected values. This approach is referred to as interface pinning, and the procedure is described by Algorithm 4.1.

(1) Set $d = \phi$

(2) Mark all nodes as unpinned

(3) Loop over elements

  (a) Check whether ϕ changes signs across the element

  (b) If ϕ does change signs

    (i) Loop over the nodes

    (ii) Mark each node as pinned for redistancing

Algorithm 4.1: Pinning nodes for redistancing

4.2 Level Set Performance

The performance of these three level set variations is evaluated on four test problems: plug flow in a cubical domain, a rotating semi-circle, an advecting sphere, and a dam break problem. The first three of these are applied velocity, red water/blue water problems, while the third is an air-water problem in which the flow is solved.

4.2.1 Performance Measures

These level set variations are evaluated on several criteria. Those criteria are the computational cost, accuracy, ease of implementation, parallelizability, volume conservation, and preservation of curvature. These are all quantitative performance measures, except for ease of implementation, which is qualitative. Not all of the performance measures can be usefully computed for all problems.

4.2.1.1 Computational Cost

The computational cost is measured using the *cpu time* function in Fortran to track the time that is spent on redistancing operations. The computational cost can then be evaluated both in terms of the time per redistancing operation and as a share of the total computational time.

4.2.1.2 Accuracy

The accuracy of each variation is calculated by comparing the computed interface location (after some number of time steps and redistancing operations) to the expected interface location (which is either known, in the applied velocity cases, or derived from experimental data, in the case of the dam break problem). Accuracy also includes the convergence of the distance field, that is, given a limit on the number of redistancing iterations for a time step, how well does the scalar field return to a distance field with a gradient of unit magnitude.

4.2.1.3 Ease of Implementation

Ease of implementation is a qualitative measure and is not dependent on the problem being solved, since once a variation has been coded, it can be applied to any problem. Ease of implementation is based on the number of routines and data structures that need to be added to the base code for each redistancing variation. Ease of implementation considers not

only the work to implement the variation for serial processing but also the work to implement it for parallel processing.

The redistancing without constraints is the simplest variation to implement. It requires a routine to set up and solve the redistancing equation, but does not require any routines or data structures that are not also required in the other two variations. This is true for parallel processing as well as serial.

The global volume constraint is the most difficult variation to implement. It requires the evaluation of two integrals of the Heaviside function at the elemental level and projection of the results to global nodes in order to find the new distance field. In parallel, this variation requires two additional communication steps: one at the beginning of the constraint solution to collect the needed information from the copy nodes, and one at the end to propagate the solution back to the copy nodes.

The variation that permits no interface motion is of intermediate difficulty to implement. It requires a simple routine to check whether the level set changes signs across an element and a means of marking the global nodes belonging to those elements so that those nodes' boundary conditions can be changed appropriately. In parallel, this variation requires all nodes on part boundaries to share their marking status, so that if any node is marked, all copies of that node are also marked. This is done through two communications: one in which the marking status is collected and one in which the correct status is propagated to all copies of a node.

4.2.1.4 Parallelizability

The parallelizability of each variation has two aspects. The first is how easily the variation can be adapted to parallel operation; this aspect is captured in the evaluation of ease of implementation. The second aspect is how each variation performs in parallel. Parallel performance is evaluated by examining the scaling behavior of each variation. For

an algorithm that scales perfectly, each doubling in the number of processors corresponds to a halving of the computational time.

4.2.1.5 Volume Conservation

Conservation of volume is evaluated by examining the relative change in one phase over the course of a simulation. The volume can be directly computed from the distance function in two ways. The first is through spatially integrating the distance function, and the second is by using geometric cut cell fractions. In this work, both options are used. Integrating the distance function over the computation domain is consistent with the way that the distance function is used to define fluid properties (i.e., through the Heavised function). Using the distance field to construct planar interface facets in the cells containing the interface and then using those facets to compute geometric volume fractions offers the simplicity of being only a geometric operation and is consistent with the volume of fluid approach, which may be advantageous for future work. The results of the two computations, in general, are within 0.2% of each other but do not agree exactly.

4.2.1.6 Preservation of Curvature

Preservation of curvature is evaluated by computing the highest interface curvature resulting from each method after some number of time steps. Although the curvature of the interface can be determined, the preservation of curvature can only be evaluated quantitatively in cases where the exact correct curvature is known (e.g., the rotating semi-circle). In cases where the correct curvature is unknown (i.e., the dam break problem), this is a qualitative performance measure.

4.2.2 Applied Velocity Tests

There are three applied velocity tests used to evaluate the level set redistancing schemes. These problems are plug flow through a cubical domain, a rotating semi-circle, and an ad-

vecting sphere. All three problems are red water/blue water problems where there are not differences in material properties between the two phases.

4.2.2.1 Plug Flow

The plug flow model is the same one used in testing the VOF and CLSVOF algorithms. It is a cubical domain with flow along the x-axis and two fluid interfaces normal to the flow direction. The model is described in detail in Section 3.1.5.1 and pictured in Figure 3.1. The three level set algorithms were evaluated on two all-tetrahedral meshes. The coarser mesh has an element size of 0.01 m (or 10 elements per model edge) for a total of approximately 7k elements. The finer mesh has an element size of 0.005 m (or 20 elements per model edge) for a total of approximately 50k elements. On each mesh, the simulation was run for one domain transport time (100 time steps of 0.001 s on the coarse mesh and 200 time steps of 0.0005 s on the fine mesh).

The three level set variations performed similarly, as expected for such a simple problem. Figure 4.1 shows the interfaces for each variation after one domain transport time and the original interface. The method allowing no interface motion during redistancing had the smallest deviation from the expected interface, while the unconstrained redistancing method had the largest. The accuracy of all three methods improved with mesh refinement.

The three level set variations also had similar results on both meshes for computational cost, measured as a percentage of the total computational time, and volume conservation, measured as a percent change from the initial volume. For this problem, the volume computed geometrically using interface facets more nearly matched the initial volume for all cases. The interface pinning variation had both the lowest computational expense and the best volume conservation. The global volume constraint had the highest computational expense on both meshes. The two volume computation methods give different results in volume conservation performance for the global volume constraint and unconstrained methods. The numeric values for these results are presented in Tables 4.1 and 4.2.

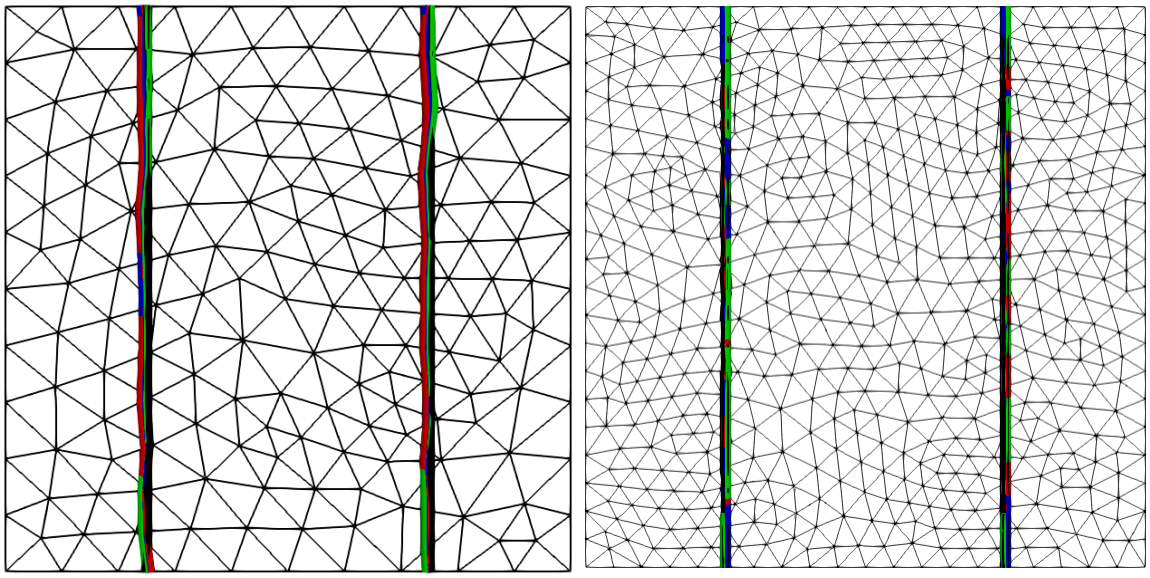


Figure 4.1: The interfaces for each of the level set variations after one domain transport time and the original interface. The coarse mesh is on the left, and the fine mesh is on the right. The interfaces from redistancing with no constraints are shown in green, the ones from the global volume constraint in blue, and the ones allowing no interface motion in red. The original interface is shown in black.

	Computational Cost	Accuracy	Volume Conservation
No Constraint	42.3	2	-0.40%, 0.06%
Global Volume Constraint	46.9	1	-0.38%, 0.16%
No Interface Motion	42.5	2	-0.26%, 0.06%

Table 4.1: Summarized performance measures for the three redistancing variations as applied to the plug flow problem. Computational cost is given as a percentage of the total computational time. The methods are ranked for accuracy (1 is most accurate). Volume conservation is a percent change; the first number is computed using the integrated distance function, and the second geometrically from interface facets.

	Computational Cost	Accuracy	Volume Conservation
No Constraint	43.1	3	-1.36%, 0.02%
Global Volume Constraint	48.2	2	-1.40%, 0.0%
No Interface Motion	42.6	1	-1.20%, 0.0%

Table 4.2: Summarized performance measures for the three redistancing variations as applied to the plug flow problem. Computational cost is given as a percentage of the total computational time. The methods are ranked for accuracy (1 is most accurate). Volume conservation is a percent change; the first number is computed using the integrated distance function, and the second geometrically from interface facets.

4.2.2.2 Rotating Semi-Circle

The rotating semi-circle model is the same one used in testing the VOF and CLSVOF algorithms. It is a thin square domain with a purely rotational applied velocity field. The model is described in detail in Section 3.1.5.2 and pictured in Figure 3.3. The three level set algorithms were evaluated on two all-tetrahedral meshes. The coarser mesh has an element size of 0.04 m (or 15 elements per semi-circle diameter) for a total of approximately 5k elements. The finer mesh has an element size of 0.025 m (or 20 elements per semi-circle diameter) for a total of approximately 10k elements. The simulation was run for one revolution of the semi-circle (314 time steps of 0.01 s).

As they did for the plug flow problem, all three level set variations performed similarly on the rotating semi-circle on both meshes. Figure 4.2 shows the interfaces for each variation after one rotation of the semi-circle and the original interface. All three variations rounded the sharp corners nearly identically on their respective meshes, with no one clearly outperforming the others in terms of either preservation of curvature or accuracy. All three methods had better performance in terms of preservation of curvature, accuracy, and volume conservation on the finer mesh.

The three variations had computational costs ranging from 39.6% of the simulation time (the unconstrained redistancing) to 44.0% of the simulation time (redistancing with the global volume constraint). As with the plug flow cases, volume conservation ranking depends on the calculation method, although the unconstrained redistancing had the poorest conservation of volume with both the integral and geometric calculations and on both meshes. On the coarse mesh, the interface pinning had the best volume conservation under both calculation methods, while on the fine mesh, the interface pinning method had the best volume conservation using the integral of the distance function to determine the volume, and the global volume constraint method had the best volume conservation using the geometric calculation. The numeric values for volume conservation are given in Tables 4.3 and 4.4,

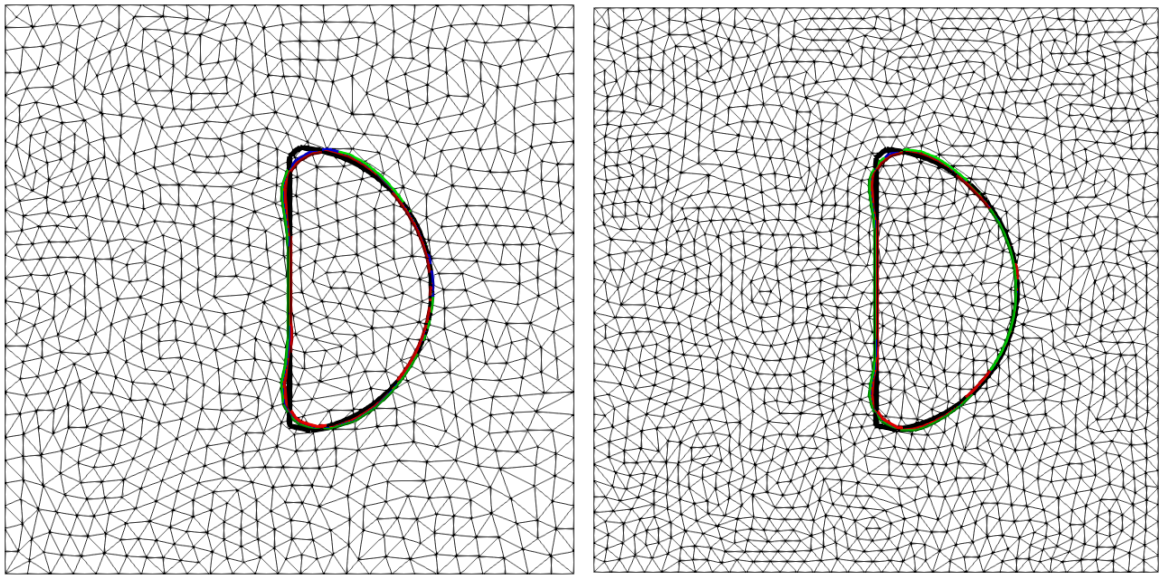


Figure 4.2: The interfaces for each of the level set variations after one revolution and the original interface. The coarse mesh is on the left, and the fine mesh is on the rigt. The interfaces from redistancing with no constraints are shown in green, the ones from the global volume constraint in blue, and the ones allowing no interface motion in red. The original interface is shown in black.

which also summarize the other performance measures. It is important to note that a key source of mass loss for this problem is an inability for the mesh, regardless of redistancing variation used, to exactly represent the sharp corners of the semi-circle. As the semi-circle rotates, those corners become more rounded (shrinking toward the concave side) because the mesh is not able to fully preserve those high curvature regions. The finer mesh is capable of accurately representing a higher curvature than the coarser mesh, and that is one of the reasons the fine mesh has generally improved volume conservation. The redistancing variations have different amounts of outward bowing of the diameter (the tendency for the semi-circle to look more like a bean) that offset the volume loss of the rounded corners that accounts for the differences in volume conservation between variations.

	Computational Cost	Accuracy	Volume Conservation	Preservation of Curvature
No Constraint	39.8	1	-0.20%, -0.21%	1
Global Volume Constraint	43.8	1	-0.15%, -0.16%	1
No Interface Motion	40.1	1	-0.04%, -0.05%	1

Table 4.3: Summarized performance measures for the three redistancing variations as applied to the rotating semi-circle problem on the coarser mesh. Computational cost is given as a percentage of the total computational time. The methods are ranked for accuracy and preservation of curvature (1 is most accurate). Volume conservation is a percent change; the first number is computed using the smooth Heaviside function, and the second geometrically from interface facets.

	Computational Cost	Accuracy	Volume Conservation	Preservation of Curvature
No Constraint	39.6	1	-0.15%, -0.13%	1
Global Volume Constraint	44.0	1	-0.15%, 0.00%	1
No Interface Motion	40.0	1	-0.09%, -0.04%	1

Table 4.4: Summarized performance measures for the three redistancing variations as applied to the rotating semi-circle problem on the finer mesh. Computational cost is given as a percentage of the total computational time. The methods are ranked for accuracy and preservation of curvature (1 is most accurate). Volume conservation is a percent change; the first number is computed using the smooth Heaviside function, and the second geometrically from interface facets.

4.2.2.3 Advecting Sphere

The advecting sphere model is the same one used in testing the VOF and CLSVOF algorithms. It is a sphere of one fluid that is advected along a rectangular domain. The

model is described in detail in Section 3.1.5.3 and pictured in Figure 3.5. The three level set algorithms were evaluated on two all-tetrahedral meshes. The coarse mesh has an element size of 0.025 m (or 6 elements per sphere diameter) for a total of approximately 10k elements. The coarse mesh has an element size of 0.0125 m (or 12 elements per sphere diameter) for a total of approximately 66k elements. The simulation was run for one domain transport time (100 time steps of 0.005 s on the coarse mesh, and 200 time steps of 0.0025 s on the fine mesh).

The three level set variations all performed similarly on the advecting sphere problem. Figure 4.3 shows that on each mesh all three variations produced nearly identical interfaces (only a cross-section of the domain is shown) after one domain transport time, and that all three experienced some stretching in the stream-wise direction. The stream-wise stretching of the sphere is more pronounced on the coarser of the two meshes. On the coarse mesh, the interface pinning method had the greatest distortion (8.80%), while the unconstrained redistancing had the least distortion (7.52%). On the fine mesh, the three methods had nearly identical distortion, with the interface pinning method having slightly less distortion than the other two methods (2.39% vs. 2.40%).

The three variations had computational costs ranging from 40.8% of the simulation time (the unconstrained redistancing on the coarse mesh) to 46.2% of the simulation time (redistancing with the global volume constraint on the coarse mesh). The unconstrained redistancing method had the poorest conservation of volume under both volume calculations and on both meshes. The interface pinning and global volume constraint methods had equivalent conservation when volume was calculated using the integral of the distance function, while the interface pinning method had the best conservation of volume as judged by the geometric volume calculation on the fine mesh. The numeric values for volume conservation are presented in Tables 4.5 and 4.6, which also summarize the other performance measures. As with the rotating semi-circle in Section 4.2.2.2, one of the factors in the volume change for this problem is the mesh's inability to perfectly represent the curvature of the sphere.

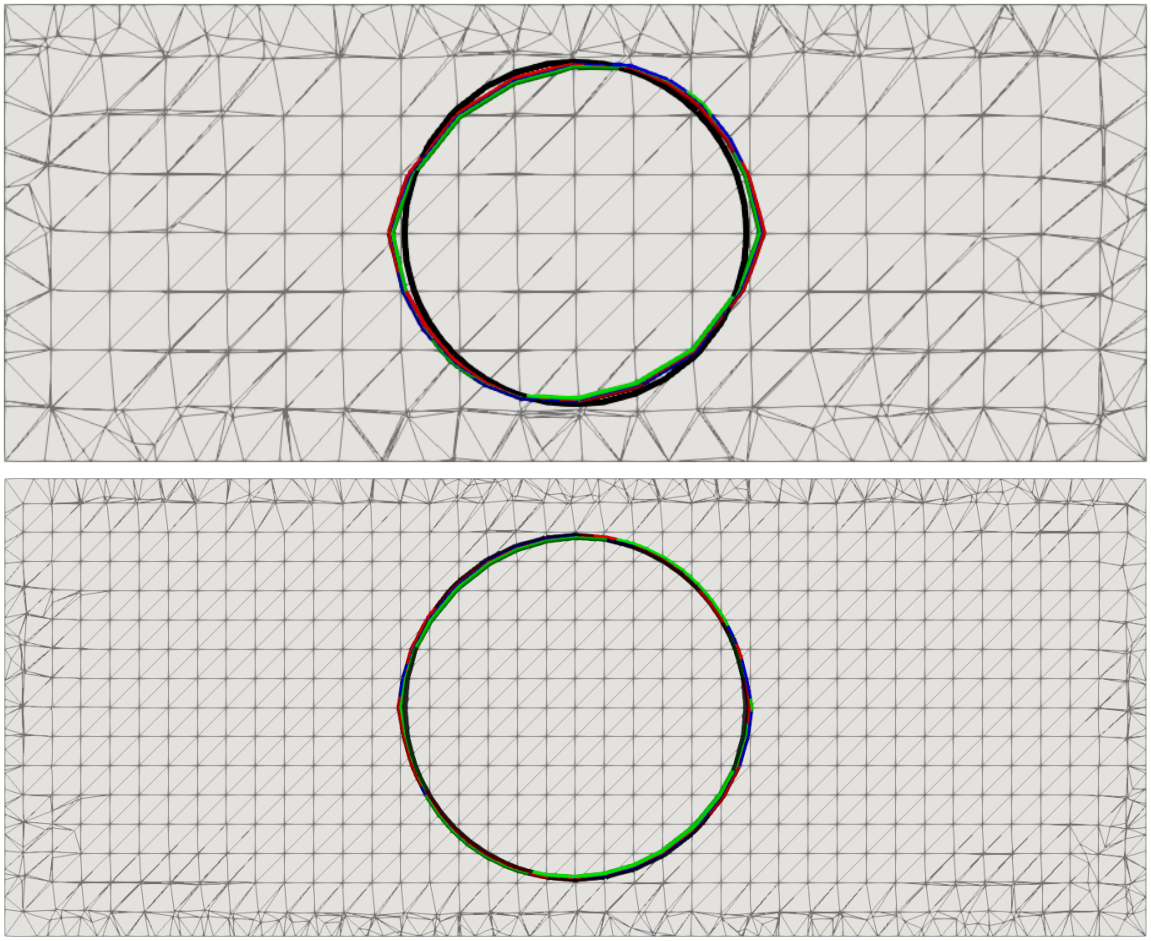


Figure 4.3: The interfaces for each of the level set variations after one domain transport time and the original interface are shown on a cross-section of the domain, with the flow direction being from left to right. The upper image is the coarser mesh, and the lower image is the finer mesh. The interfaces from redistancing with no constraints are shown in green, the ones from the global volume constraint in blue, and the ones allowing no interface motion in red. The original interface is shown in black.

In this case, however, stretching of the sphere in the stream-wise direction dominates loss of curvature in the same direction and amplifies the curvature loss in the other directions, leading to a net gain of volume for the sphere.

	Cost	Accuracy	Volume Conservation
No Constraint	40.8	92.48%	0.66%, 0.63%
Global Volume Constraint	46.2	91.92%	0.45%, 0.45%
No Interface Motion	41.0	91.20%	0.45%, 0.45%

Table 4.5: Summarized performance measures for the three redistancing variations as applied to the advecting sphere problem on the coarse mesh. Computational cost is given as a percentage of the total computational time. Accuracy is given as one minus the percent elongation of the interfaces along the centerline of the domain. Volume conservation is a percent change; the first number is computed using the integral of the smooth Heaviside function, and the second geometrically from interface facets.

	Cost	Accuracy	Volume Conservation
No Constraint	41.3	97.60%	0.172%, 0.171%
Global Volume Constraint	46.4	97.60%	0.008%, 0.016%
No Interface Motion	41.5	97.61%	0.008%, 0.009%

Table 4.6: Summarized performance measures for the three redistancing variations as applied to the advecting sphere problem on the fine mesh. Computational cost is given as a percentage of the total computational time. Accuracy is given as one minus the percent elongation of the interfaces along the centerline of the domain. Volume conservation is a percent change; the first number is computed using the integral of the distance function, and the second geometrically from interface facets.

4.2.3 Dam Break

The dam break is a quasi-two-dimensional air/water problem in which water is initially confined by a membrane to a box at one end of the domain. At time $t = 0$ the membrane breaks and the column of water begins to collapse. The computational domain,

shown in Fig. 4.4, is based on experiments by Martin and Moyce [50] to enable direct comparison to both their experimental data and previous computational results generated using PHASTA [65, 66]. In addition to having phases with different material properties, the dam break problem includes surface tension, gravity, and the solution of the incompressible Navier-Stokes equations, rather than an applied velocity field.

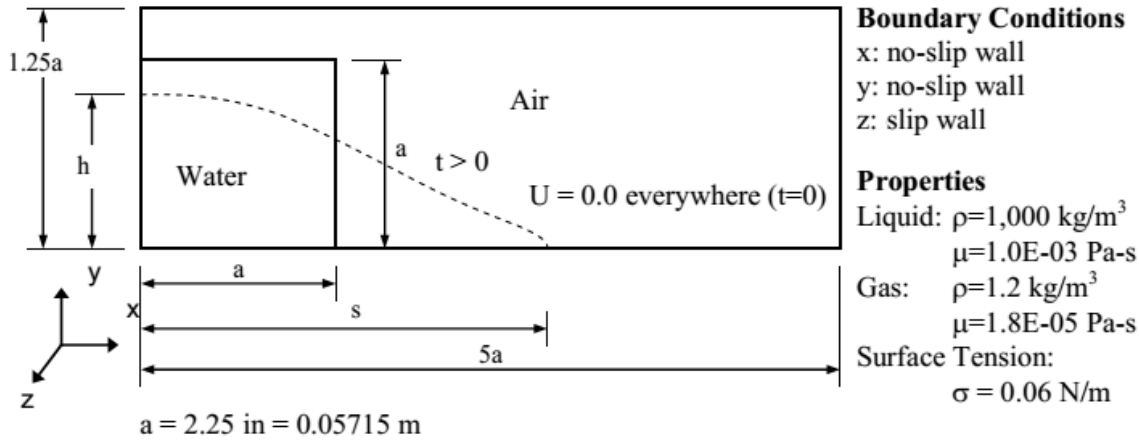


Figure 4.4: The problem domain for a quasi-two-dimensional dam break. The column of water is initially confined to the box at the left; starting at time $t = 0$, the column of water collapses. Image modified from one created by J. Rodriguez [65]; used with permission.

The three level set variations were initially evaluated on an all-tetrahedral mesh with an element size of approximately 0.005 m (or about 15 elements for the initial dam break height) for a total of approximately 15k elements. Each variation was run with the same tolerance on the redistancing calculation and the same limit (30) on the number of redistancing iterations. The simulation was run long enough for the surge front to approach but not contact the far wall (700 time steps of 0.0003s).

The three variations produced similar interfaces, as can be seen in Figure 4.5, which shows the interfaces for each variation at three different times. The interfaces from the simulations with the global volume constraint and no interface motion during redistancing both had regions of higher curvature than the interface from unconstrained redistancing. They also both maintained larger portions of the interface along the left and bottom walls of

the domain than the unconstrained redistancing did, with the no interface motion variation maintaining the most interface along the bottom wall.

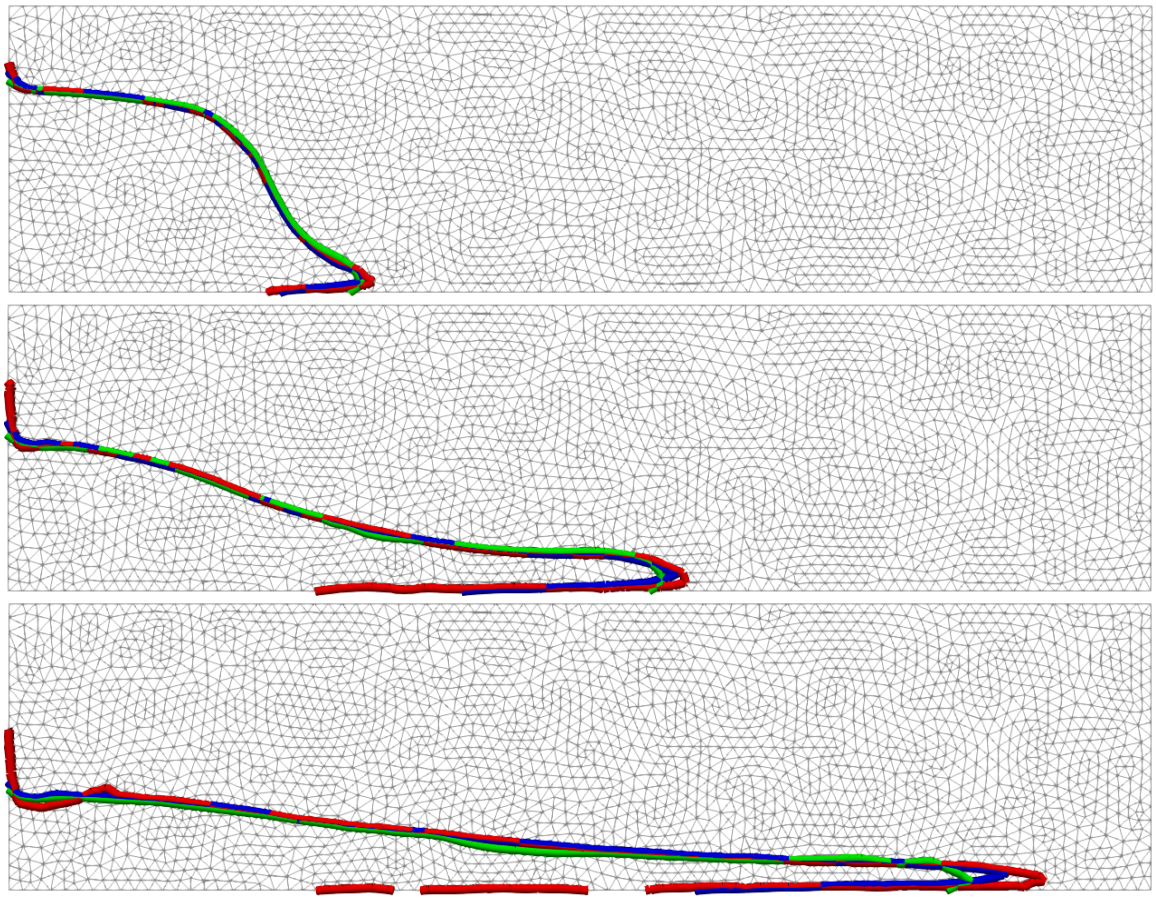


Figure 4.5: The evolution of interfaces for the dam break problem. From top to bottom, the interfaces are shown at 0.06 s, 0.135 s, and 0.21 s. The interface from redistancing with no constraints is shown in green, the one from the global volume constraint in blue, and the one allowing no interface motion in red.

Figure 4.6 and figure 4.7 show how the column heights and surge front locations compare to those measured experimentally by Martin and Moyce [50]. For all three level set variations, the column collapses slightly faster than for the experiment, and the computed surge fronts slightly lead the experimental surge front, especially for the global volume constraint and no interface motion variations. However, all three variations show generally good agreement with the experimental results. For the global volume constraint and no interface

motion variations, the faster moving surge front is likely due to the lack of a surface wetting model on the no-slip bottom face of the domain. The no-slip boundary condition there is better met by those two variations, but that means that their surge fronts move on very thin layers of air, with lower friction.

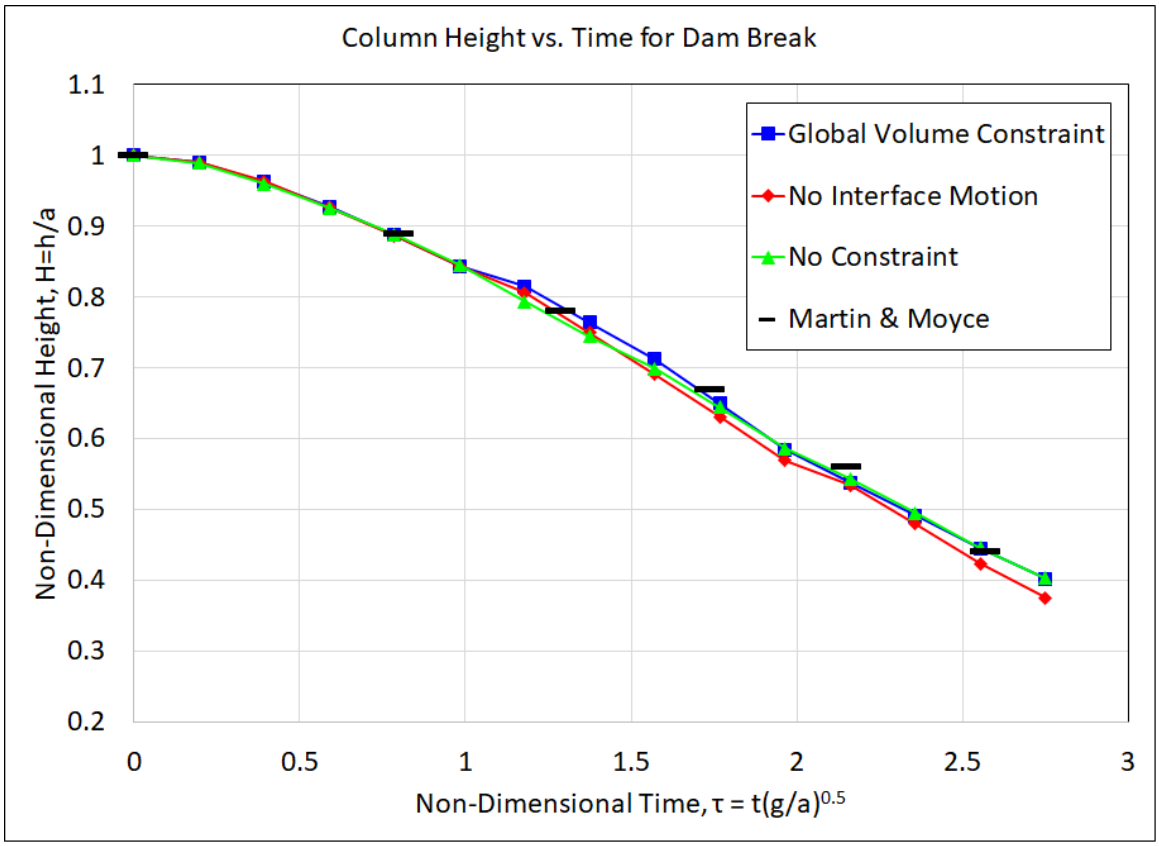


Figure 4.6: The non-dimensional column heights are shown plotted versus non-dimensional time for each of the three level set variations and for Martin and Moyce's experimental results.

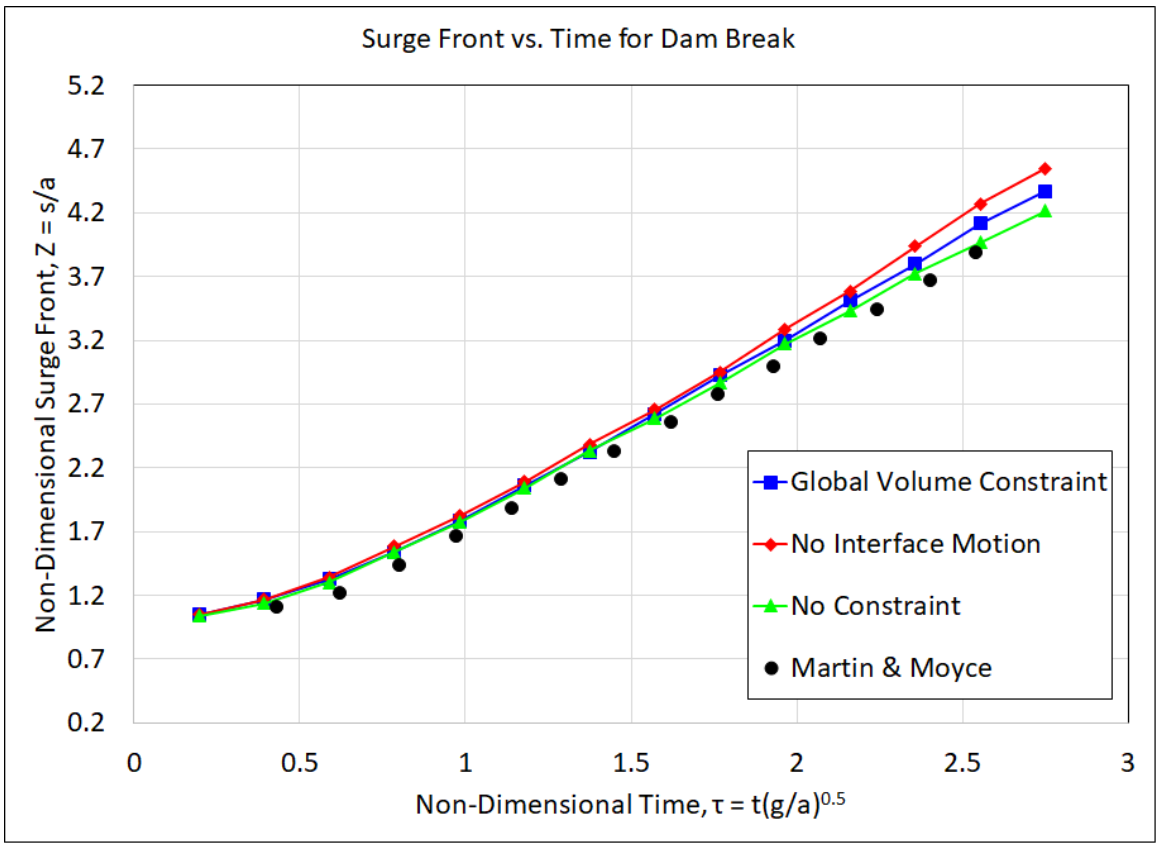


Figure 4.7: The non-dimensional surge front lengths are shown plotted versus non-dimensional time for each of the three level set variations and for Martin and Moyce's experimental results.

The interfaces shown in figure 4.5 are the cumulative product of redistancing at each time step, and therefore diverge from each other as the simulations progress. To more closely examine the behavior of each level set redistancing variation, it is useful to look at redistancing after a single time step. Two different cases were used to examine redistancing behavior for a single time step. The first case started from an intermediate time step (500) from the simulation run initially with a global volume constraint on redistancing (because it has been considered the standard level set redistancing variant), and the second case started from the same intermediate time step from the simulation initially run without redistancing constraints (because unconstrained redistancing appears to have a different equilibrium than the other two variations). In each case a single large time step (0.003s) was taken,

and after solving the flow and first level set scalar, the second scalar was solved using each of the variations. Each variation was run for 200 redistancing iterations. Figure 4.8 shows the three resulting interfaces (the interface before redistancing is identical to the interface from redistancing allowing no interface motion and so is not visible) for each starting case. In each case, both the unconstrained redistancing and the redistancing with global volume constraint lost some curvature in the surge front, with the global volume constraint redistancing preserving the high curvature region better than the unconstrained redistancing. Additionally, both the unconstrained redistancing and the redistancing with global volume constraint had changes in volume: for the case starting from a global volume constraint simulation, redistancing with a global volume constraint had a 0.004% loss in volume and the unconstrained redistancing had a 1.53% gain in volume, as calculated using the geometric volume fraction; for the case starting from an unconstrained redistancing simulation, redistancing with a global volume constraint had a 0.005% loss in volume and the unconstrained redistancing had a 0.084% loss in volume, as calculated using the geometric volume fraction. In both cases, all three methods showed no change in volume as calculated by integrating the smooth Heaviside function. The large change in volume for unconstrained redistancing for the case starting with an interface from the global volume constraint simulation is due to unconstrained redistancing variation having a different equilibrium state for the interface (a result of seeking a unit gradient everywhere without restricting interface motion) rather than to inherently poor performance of the method that always results in such large volume changes during redistancing.

Convergence of redistancing in both cases was set as a residual error of no more than $1*10^{-10}$, and a fully converged redistancing operation should yield a unit gradient everywhere in the domain. None of the three variations in either case converged within 200 iterations. For the case starting from a global volume constraint simulation, the unconstrained redistancing reached a residual error of $6.929*10^{-10}$; redistancing with a global volume constraint reached a residual error of $9.522*10^{-10}$; and redistancing with no interface motion reached a residual

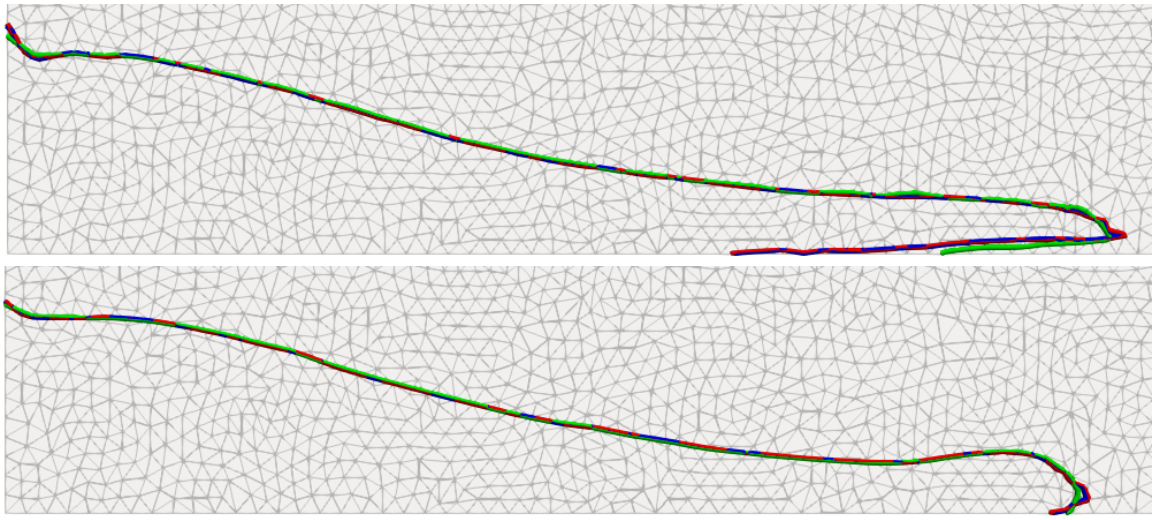


Figure 4.8: The upper image shows the interfaces resulting from redistancing after a single time step starting from a global volume constraint simulation; the lower image shows the same starting from a simulation with no redistancing constraints. In each case, the three interfaces resulting from each of the variations differ after a single time step. The interface from redistancing with no constraints is shown in green, the one from the global volume constraint in blue, and the one allowing no interface motion in red. The interface before redistancing is exactly coincident with the interface from redistancing with no interface motion and therefore not visible.

error of $6.626 * 10^{-10}$. Figure 4.9 shows the resulting gradient fields (as $1 - |\nabla\phi|$) for each redistancing variation for the global volume constraint case. For the case starting from an unconstrained redistancing simulation, the unconstrained redistancing reached a residual error of $5.532 * 10^{-10}$; redistancing with a global volume constraint reached a residual error of $5.851 * 10^{-10}$; and redistancing with no interface motion reached a residual error of $5.601 * 10^{-10}$. Figure 4.10 shows the resulting gradient fields (as $1 - |\nabla\phi|$) for each redistancing variation for the unconstrained redistancing case. In each case, all three variations have a region where the gradient is zero in the surge front, where there are interfaces both above and below the center of the surge front. The unconstrained redistancing otherwise produces a level set field where the gradient is most nearly one everywhere, including in the near vicinity of the interface. In the case starting from a global volume conservation simulation, both the global volume constraint redistancing and the redistancing with no interface motion have an

area of higher gradient within approximately one element of the interface. Redistancing with no interface motion resulted in the highest gradient areas of any of the three variations, but it also had more areas along the interface with a gradient at or near one than redistancing with the global volume constraint.

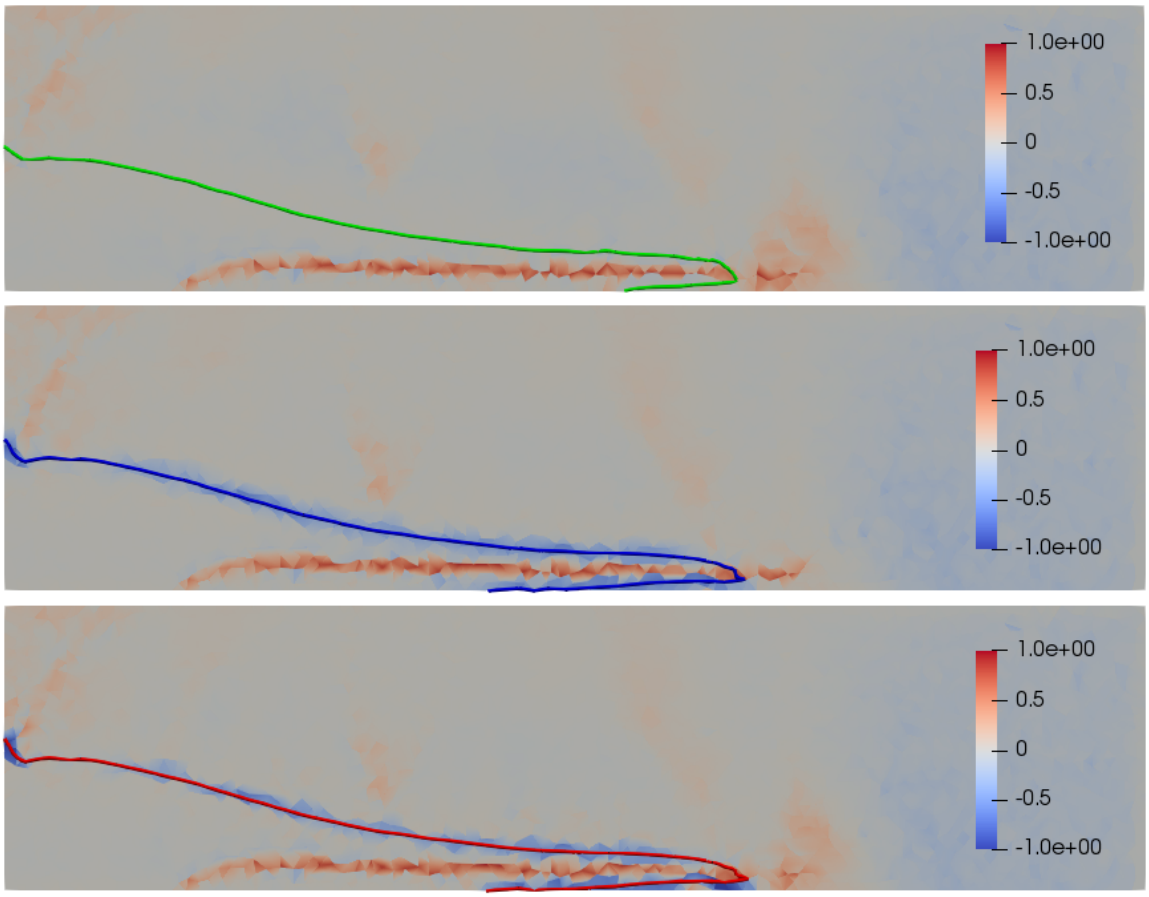


Figure 4.9: The interfaces and level set gradient fields after redistancing are shown (from top to bottom) for unconstrained redistancing, redistancing with global volume constraint, and redistancing with no interface motion for the case starting from a global volume constraint simulation.

Computational cost can be looked at two different ways for the dam break problem. It can be viewed as a percent of the total computational time for the entire simulation, while redistancing over a single time step provides a useful measure of time per redistancing iteration. By both measures, the unconstrained redistancing and redistancing with no interface motion had approximately equivalent computational costs while redistancing with the

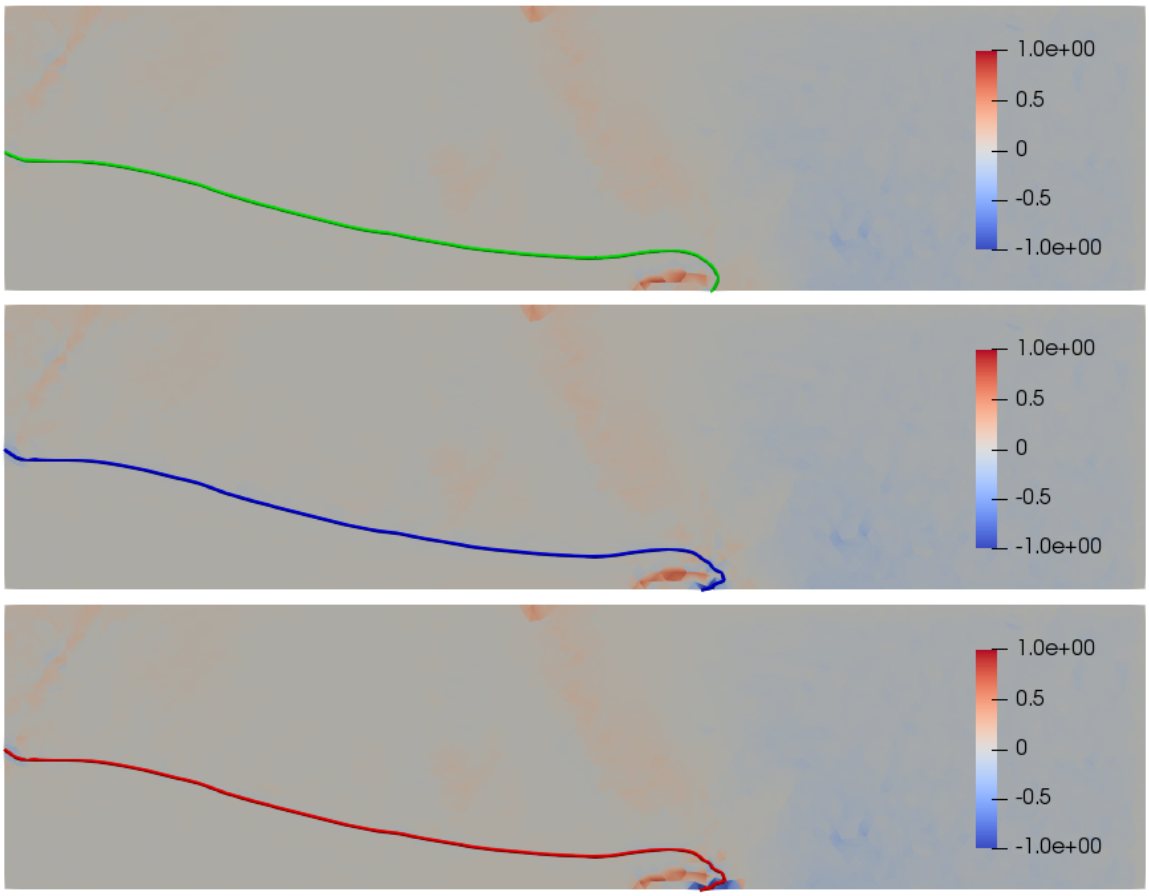


Figure 4.10: The interfaces and level set gradient fields after redistancing are shown (from top to bottom) for unconstrained redistancing, redistancing with global volume constraint, and redistancing with no interface motion for the case starting from an unconstrained redistancing simulation.

global volume constraint had higher computational costs. Tables 4.7 and 4.8 summarize the performance measures for the three redistancing variations applied over a single time step for the cases starting from a global volume constraint simulation and from an unconstrained redistancing simulation, respectively.

It is important to note that volume conservation behavior is not monotonic over the course of a simulation: on some time steps, the volume increases, while on others it decreases. Fig. 4.11 shows the behavior of the volume fraction over the course of 0.21 s of simulation for each redistancing variation. For much of the simulation, unconstrained redistancing has poorer volume conservation than either redistancing with a global volume constraint or

	Computational Cost	Volume Conservation	Preservation of Curvature	Accuracy
No Constraint	0.11 s	1.52%	3	$6.929 * 10^{-10}$
Global Volume Constraint	0.13 s	0.004%	2	$9.522 * 10^{-10}$
No Interface Motion	0.11 s	0.0%	1	$6.626 * 10^{-10}$

Table 4.7: Summarized performance measures for the three redistancing variations as applied to a single time step of the dam break problem for the case starting from a global volume conservation simulation. Computational cost is given as seconds per redistancing iteration. Volume conservation is a percent change computed using geometric volume fractions. Preservation of curvature is ranked from best (1) to worst (3). Accuracy is given as the residual error.

	Computational Cost	Volume Conservation	Preservation of Curvature	Accuracy
No Constraint	0.18 s	-0.084%	3	$5.532 * 10^{-10}$
Global Volume Constraint	0.19 s	-0.005%	2	$5.851 * 10^{-10}$
No Interface Motion	0.16 s	0.0%	1	$5.601 * 10^{-10}$

Table 4.8: Summarized performance measures for the three redistancing variations as applied to a single time step of the dam break problem for the case starting from an unconstrained redistancing simulation. Computational cost is given as seconds per redistancing iteration. Volume conservation is a percent change computed using geometric volume fractions. Preservation of curvature is ranked from best (1) to worst (3). Accuracy is given as the residual error.

redistancing with no interface motion, but looking only at the end of the simulation suggests that unconstrained redistancing has the best volume conservation, with a loss of 1.15% compared to gains of 1.84% and 1.23% for the global volume constraint and no interface

motion redistancing, respectively. Table 4.9 summarizes the performance measures for the three redistancing variations over the entire dam break simulation.

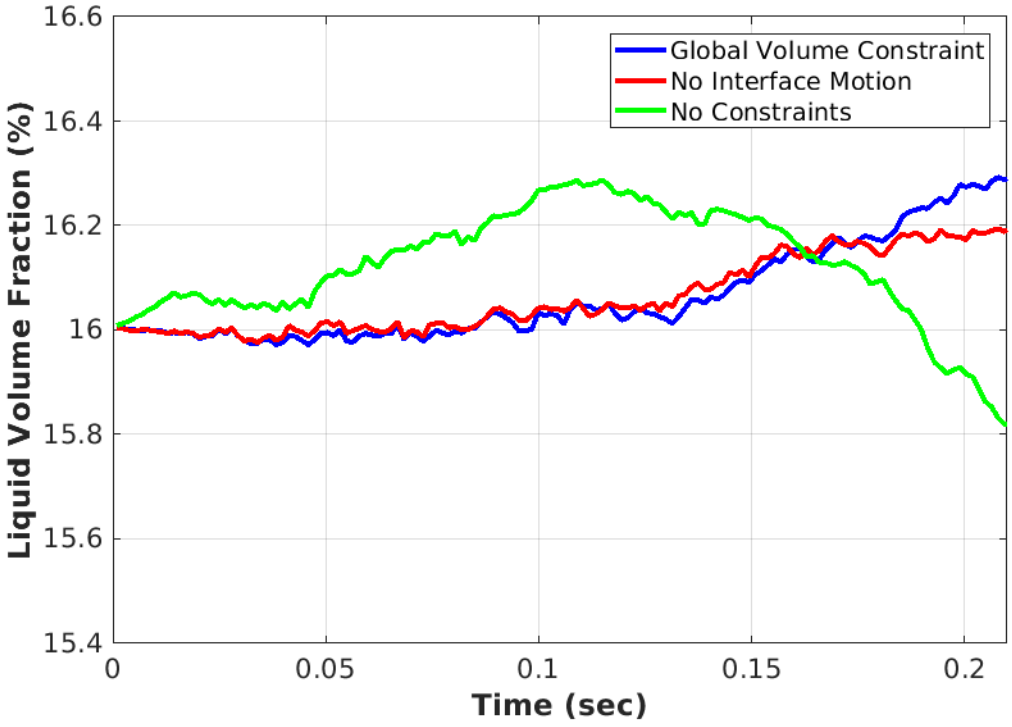


Figure 4.11: The history of the volume fraction for each redistancing variation over the course of a 0.21 s simulation. The unconstrained redistancing variation is shown in green, the global volume constraint in blue, and redistancing with no interface motion in red.

Overall, considering both the single time step and whole simulation performance of the three redistancing variations, redistancing with no interface motion offers an improvement in computational cost and in preservation of curvature, as well as volume conservation, over redistancing with the global volume constraint. The simulation with unconstrained redistancing, however, most closely matched the experimental results. Although the unconstrained redistancing had the best volume conservation at the last time step of the simulation, the volume histories for the simulations show that there is not clearly one method that is best for overall volume conservation.

	Computational Cost	Accuracy	Volume Conservation	Preservation of Curvature
No Constraint	74.6%	1	-1.15%	3
Global Volume Constraint	78.3%	2	1.84%	2
No Interface Motion	74.9%	3	1.23%	1

Table 4.9: Summarized performance measures for the three redistancing variations as applied to the complete dam break problem simulation. Computational cost is given as a percentage of the total computational time. Accuracy is ranked based on comparison to experimental results. Volume conservation is a percent change computed using the integrated distance function.

4.2.4 Conclusions

Based on these four problems, redistancing with a global volume constraint performs better than unconstrained redistancing, but at a significant increase in computational expense, consistent with previous resuts (e.g., Sussman et al. [78] and Sussman and Fatemi [79]). Redistancing with no allowed interface motion offers reduced computational expense and improved preservation of curvature, compared to redistancing with a global volume constraint. Table 4.10 ranks each of the three redistancing variations from best to worst for each of the performance measures. Overall, unconstrained redistancing has the poorest performance, while redistancing with no interface motion offers a slight edge over redistancing with a global volume constraint.

	Computational Cost	Accuracy	Ease of Implementation	Parallelizability	Volume Conservation	Preservation of Curvature
No Constraint	1	3	1	1	3	3
Global Volume Constraint	3	2	3	3	2	2
No Interface Motion	2	1	2	2	1	1

Table 4.10: Summarized performance measures for the three redistancing variations. The three variations are ranked from best (1) to worst (3) for each performance measure.

Chapter 5

Interface-Based Adaptivity

An alternative (or complement) to seeking improved results through methodological improvement is to seek improved results through mesh refinement. However, computations on uniformly refined meshes can frequently become prohibitively expensive. Adaptive mesh refinement (AMR) allows for a mesh to be fine only where needed or desired, as indicated by solution errors or some feature of the flow. When applied iteratively during a flow solution, AMR can refine coarse parts of the mesh and coarsen fine parts of the mesh where refinement is no longer necessary to minimize the computational expense of the flow solution. AMR cycles have their own computational expense, which means that the computational savings in the flow solution steps must be balanced against the computational expense of the AMR cycles.

This section examines the use of interface information to drive adaptation cycles. AMR is applied to a dam break problem and to an annular flow problem. Both problems are run on parallel adaptive meshes.

5.1 Adaptation Method

Historically, parallel mesh adaptation has relied on using the file system for storing each adapted mesh. For example, SCOREC/core is a parallel mesh adaptation tool that works with PHASTA. It takes as inputs a parallel mesh and any of several error indicators and produces as new files a new parallel mesh that is refined where the error was high and

coarsened where the error was low [34, 66]. SCOREC/core has been shown to be an effective adaptivity tool in single-phase flows.

This reliance on the file system for adaptation becomes problematic for massively parallel simulations where file system bandwidth is a performance bottleneck. To address this, Smith et al. [76] have developed a framework for carrying out the mesh adaptation using in-memory interactions in lieu of file system interactions. Their system is used in this work.

For two-phase flows, the area for which the greatest mesh refinement is desired is the area around the interface. There are two options for refinement indicators (the "error" indicators provided to SCOREC/core do not actually need to be errors) that are available to two-phase flow simulations that are not options for single-phase flow simulations. The first is proximity to the interface, and the second is the curvature of the level set.

Proximity to the interface is easily determined using the level set field, since the level set is the signed distance to the interface. The refinement band or bands are set according to

$$|\phi| < n\Delta x_0, \tag{5.1}$$

where n is the number of originally sized elements over which to refine (i.e., the thickness of the refinement band) and Δx_0 is the average element edge length of the original mesh.

Areas to be refined based on curvature are defined by the following level set limited function:

$$|\frac{(\kappa(\phi))^2}{\phi + 10^{-5}}| > E_{thresh}, \tag{5.2}$$

where E_{thresh} is a user-defined value that determines how large a region is considered as needing refinement due to having high curvature. The level set field can have high curvature away from interfaces where it has local minima or maxima. To prevent unnecessary refinement, there is an added constraint that high curvature regions must also be close to an interface in order to be marked for refinement.

The level of refinement for each band or area is determined by a user defined refinement factor that is based on edge refinement. For example, a factor of two means that each edge of the elements marked for refinement will be divided in half, resulting in eight new elements for each original element. Refinement factors should be powers of two, to be consistent with SCOREC/core's algorithms.

The refinement indicator field (labeled as an "error" field) that is passed from PHASTA to SCOREC/core is a nodal field. Where no refinement is indicated, it has the value Δx_0. Where refinement is indicated, it takes on the value $\Delta x_0/R_f$ where R_f is the refinement factor. SCOREC/core then refines the elements with edge lengths larger than the refinement indicator and coarsens elements with edge lengths smaller than the refinement indicator.

5.2 Dam Break

To assess the ability of adaptivity to improve mass conservation, the dam break problem described in Section 4.2.3 was repeated for two additional mesh systems. The first is a uniformly refined mesh, and the second is an iteratively adapted mesh with a band of refined elements centered on the interface. The uniformly refined mesh is created by starting with the mesh used in Section 4.2.3 and refining it by a factor of four (each element edge is divided by two, twice); this mesh has approximately 981k elements.

Figure 5.1 shows example interfaces for each of the redistancing variations on the fine uniform mesh. Each image shows the interface in white against background shading of fluid velocities. Each interface shows small anomalies, with the unconstrained redistancing (top image) and interface with no interface motion (bottom image) showing irregularities near the left wall of the domain, and the redistancing with a global volume constraint (center image) and redistancing with no interface motion cases showing a "bump" in the surge front. The irregularities near the left wall of the domain affect the column height, as shown in Figure 5.2, where both the unconstrained redistancing and the redistancing with no interface motion cases exhibit non-smooth progressions with time and significant deviation from the

experimental results. The case of redistancing with a global volume constraint, by contrast, behaves well and closely matches the experimental results. For the surge front, shown in Figure 5.3, all three redistancing variations agreed quite closely with each other and showed greater surge front motion than the experimental results. The greater surge front motion is due to the lack of a wall contact model in the simulation, so the surge front moves over a thin layer of air as the no-slip boundary condition is maintained on the bottom surface of the domain.

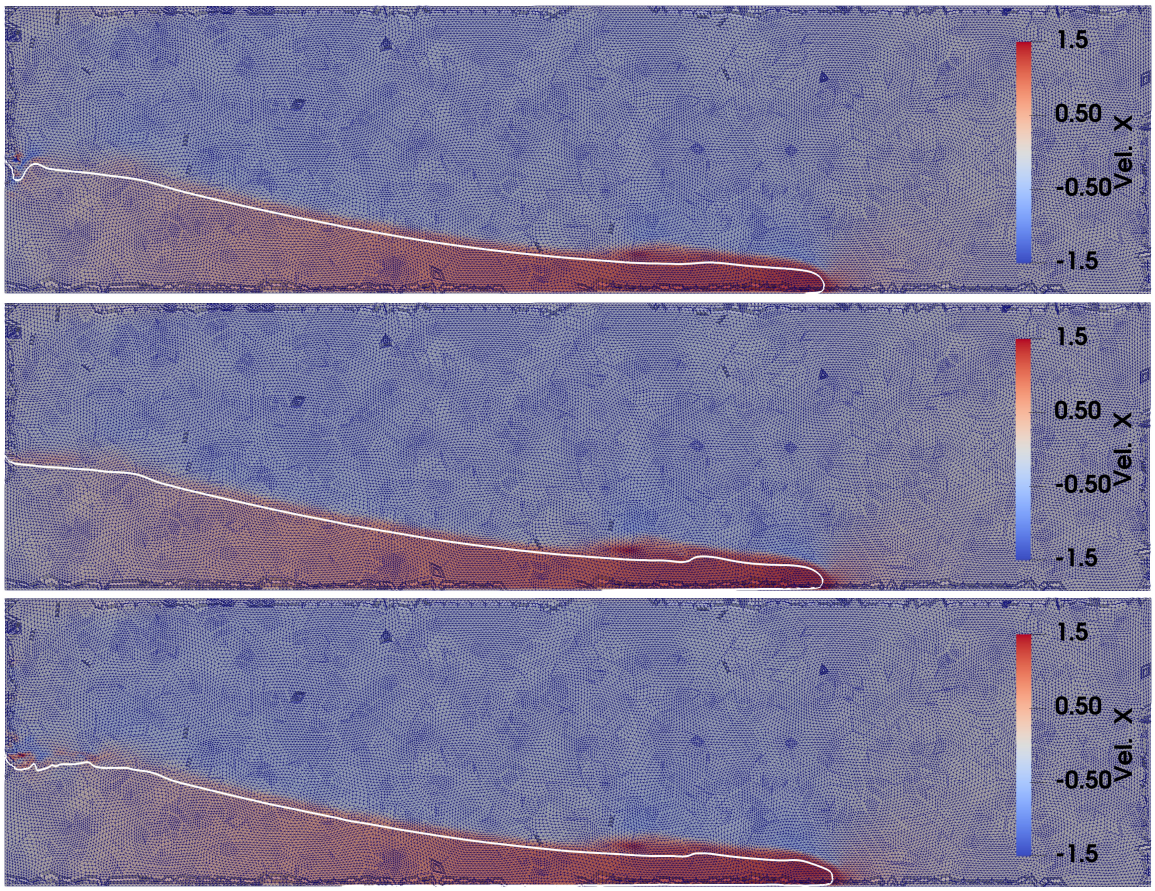


Figure 5.1: Example interfaces for the dam break simulation on a uniformly refined mesh at t=0.165 s (time step 2200). From top to bottom are the interfaces from unconstrained redistancing, redistancing with a global volume constraint, and redistancing with no interface motion.

Figure 5.4 shows the volume history for each simulation on the fine uniform mesh. All three redistancing variations had better volume conservation performance than on the

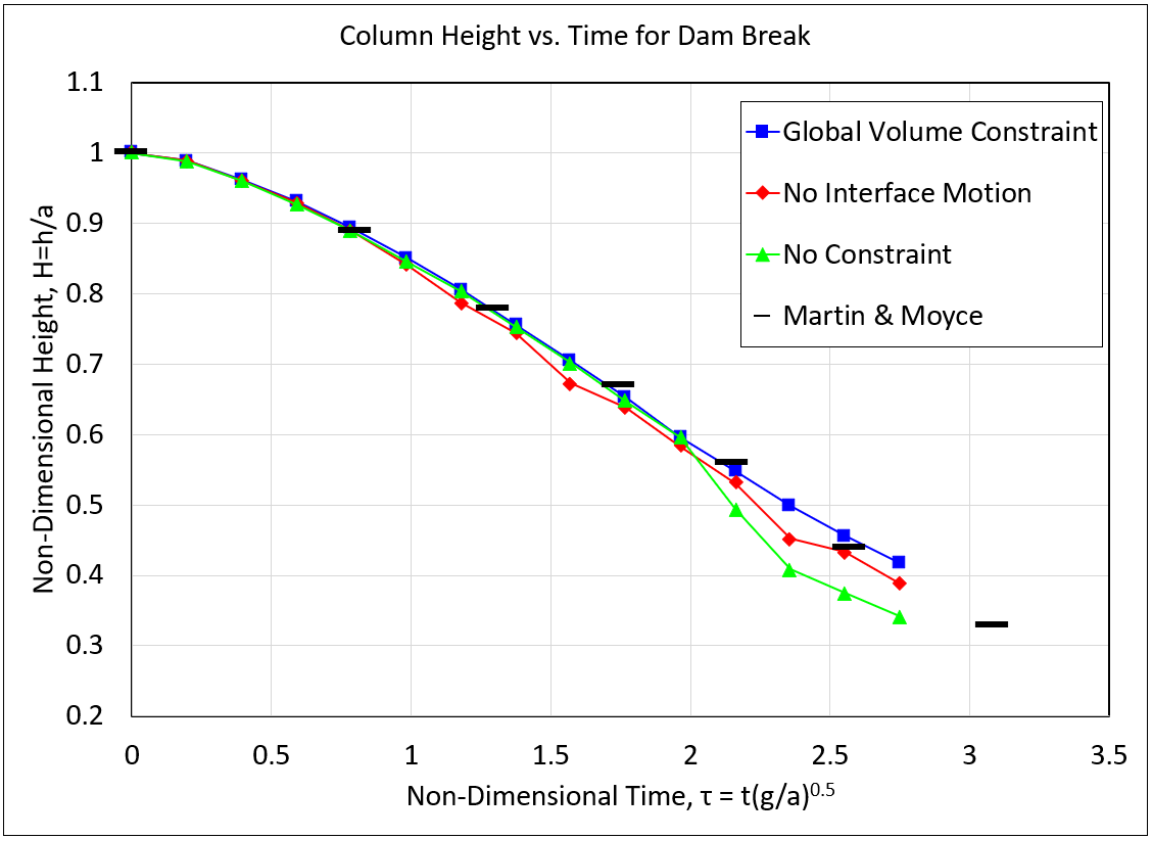


Figure 5.2: The non-dimensional column heights are shown plotted versus non-dimensional time for each of the three level set variations and for Martin and Moyce's experimental results.

coarse mesh, with improvements of roughly a factor of three. At the end of the simulation, unconstrained redistancing had a 0.37% change in volume, redistancing with a global volume constraint had a 0.42% change in volume, and redistancing with no interface motion had the poorest volume conservation performance, with a -0.51% change in volume of the liquid phase.

The enhanced volume conservation performance comes at a computational expense. The simulations on the fine mesh take longer per time step and have more time steps than the simulation on the coarse mesh. Table 5.1 gives the average time per step for each redistancing variation for both the fine and coarse meshes. In addition to the fine mesh simulations taking three to four times longer per time step, the fine mesh simulations also

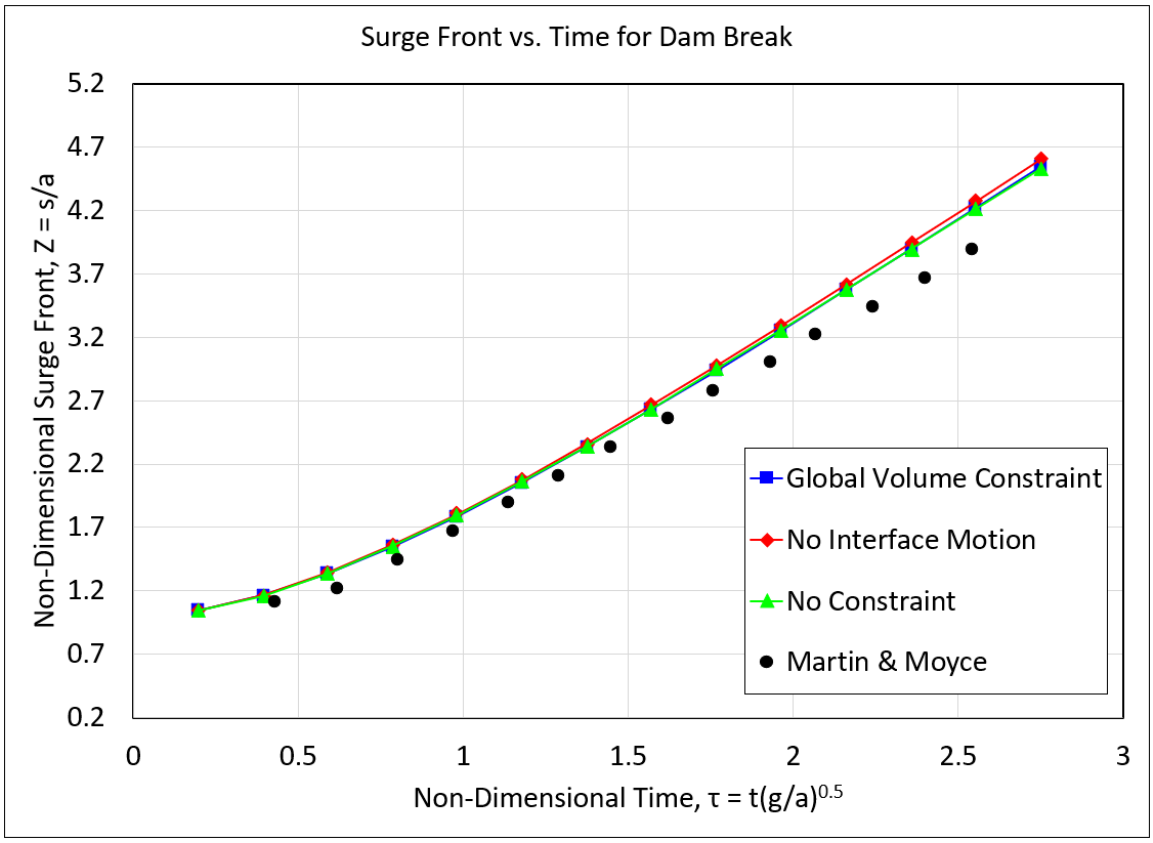


Figure 5.3: The non-dimensional surge front lengths are shown plotted versus non-dimensional time for each of the three level set variations and for Martin and Moyce's experimental results.

have four times as many time steps, so the total increase in computational time for the fine mesh simulations relative to the coarse mesh simulations is approximately a factor of twelve to sixteen.

	No Constraint	No Interface Motion	Global Volume Constraint
Coarse Mesh	0.719306	0.72033	0.45808
Fine Mesh	2.052128	2.07257	2.30005

Table 5.1: The average runtime (in seconds) per time step is given for each redistancing variation on the coarse and fine uniform meshes.

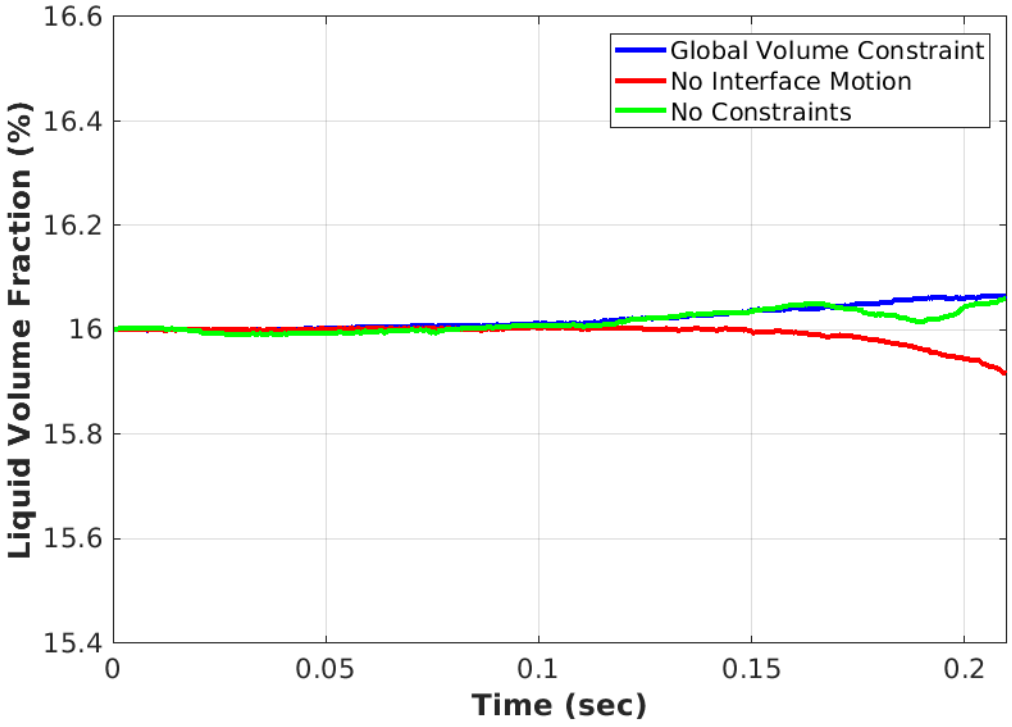


Figure 5.4: The history of the volume fraction for each redistancing variation over the course of a 0.21 s simulation on a uniformly refined mesh. The unconstrained redistancing variation is shown in green, the global volume constraint in blue, and redistancing with no interface motion in red.

An alternative to uniform refinement is to refine the mesh only over a band of elements centered about the interface. The elements within the band are refined by a factor of four (as with the uniform fine mesh), while the elements outside that band are the same size as the course mesh used in Section 4.2.3. This mesh follows the adaption strategy described above in Section 5.1, where the flow is solved for 100 time steps (of size 7.5 $ast10^{-5}$ s) between adaptations. The width of the refined band is large enough to ensure that the interface does not pass out of the band within each flow solution portion of the solve/adapt cycle. As noted above, in addition to writing the flow at the end of every 100 steps, the last interface location is used to define the new location for the band as a requested size field for the adaptation software. This solution field and size field are passed directly (using streams) to the adaptation software which then coarsens the elements that are no longer within the

refinement band and refines the elements that now fall within it. The new mesh with an interpolated solution is then passed back to the solver so that it can continue with high resolution in the band around the interface.

The progression of adapted meshes for the case with the global volume constraint is shown in Figure 5.5. Each image shows the interface in white against a background of mesh edges showing the refined band around the interface and shading showing the fluid velocity. As with the cases on the uniformly refined mesh, the interface does show some small irregularities, but in this case is generally well behaved. By contrast, the adaptive mesh cases for unconstrained redistancing and for redistancing with no interface motion behaved quite poorly (see Figure 5.6 for an example from the simulation with redistancing with no interface motion), and are therefore not included in the analysis of the adaptive meshing. Possible reasons for the poor behavior include under-resolved flow and incorrect interface thickness; this is discussed further in Section 6.2.

As shown in Figures 5.7 and 5.8, the adaptive mesh simulation and fine mesh simulation for redistancing with a global volume constraint produced nearly identical results in terms of column height and surge front position with time. Compared to the coarse mesh simulation, both refined meshes had greater overshoot on the surge front location. This is, as discussed previously, due to the no-slip boundary condition on the bottom surface of the domain allowing the surge front to move on a thin layer of air.

The adaptive mesh and uniformly refined mesh also had similar performance in volume conservation. Figure 5.9 shows the volume fraction histories for the global volume constraint simulations on each of the three meshes. Both of the simulations on refined meshes have markedly better volume conservation than the simulation on the coarse mesh.

From the above, it is clear that adapting the mesh around the interface can provide results similar to a uniformly refined mesh, in terms of both overall results and improved mass conservation. The adaptive mesh strategy additionally offers improvements in computational cost over the uniformly refined mesh. The solver portion of the adaptive mesh solution took

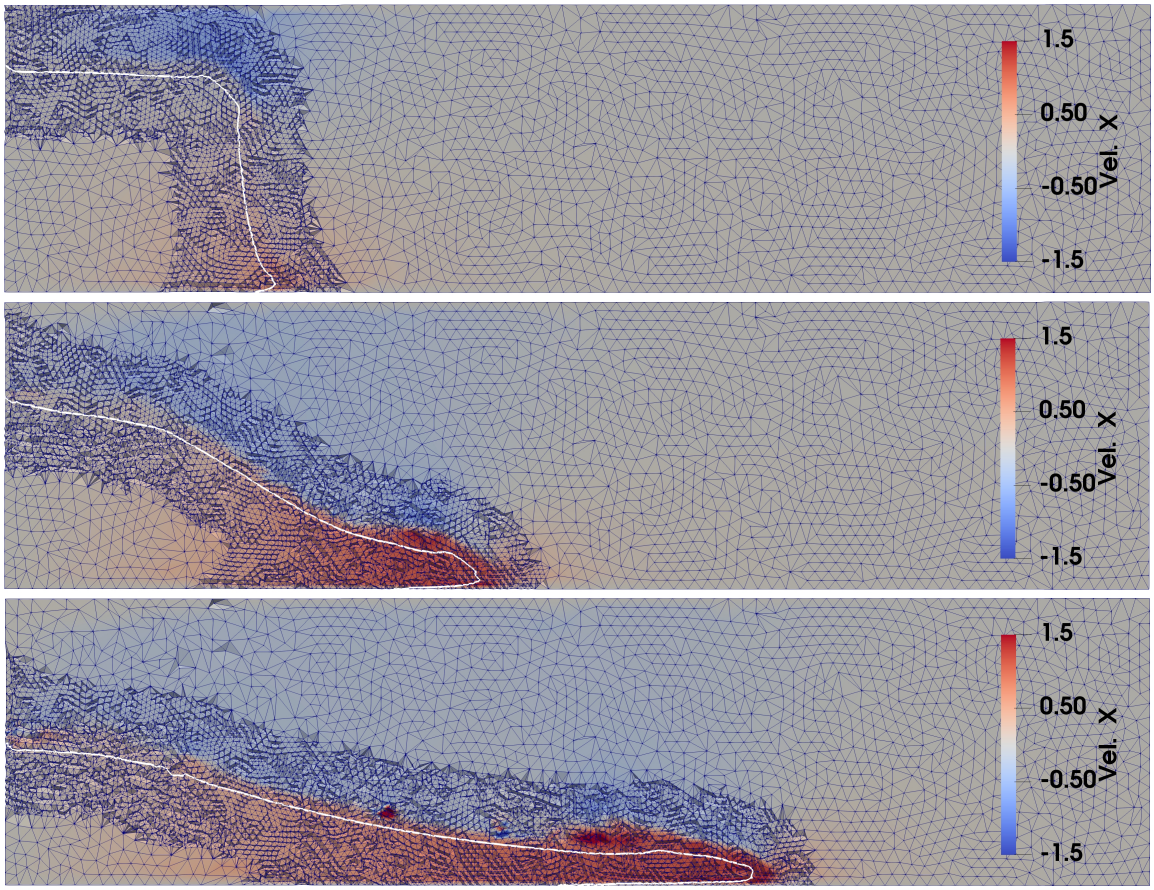


Figure 5.5: The evolution of the adaptive mesh and dam break interface for redistancing with a global volume constraint. From top to bottom, the mesh and interface are shown at 0.0315 s, 0.0915 s, and 0.1515 s.

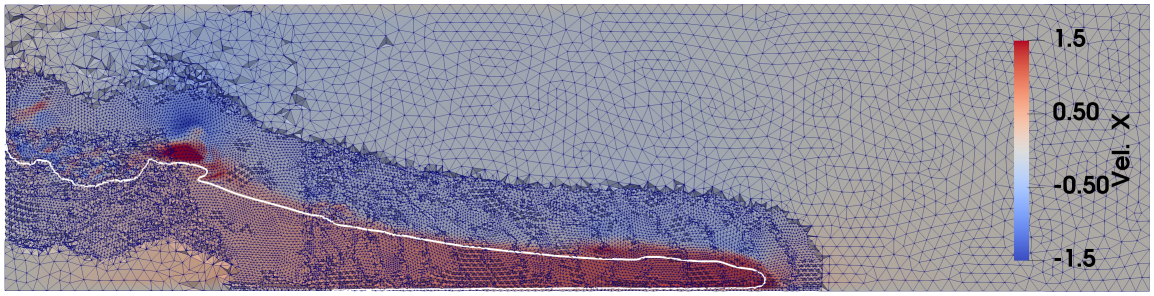


Figure 5.6: The dam break interface and adaptive mesh for the simulation with no interface motion during redistancing. This interface is from 0.15 s, and shows that the interface has lost cohesion near the left wall of the domain.

an average of 0.886 s per time step, and each adaptation portion took an average of 6.231 s.

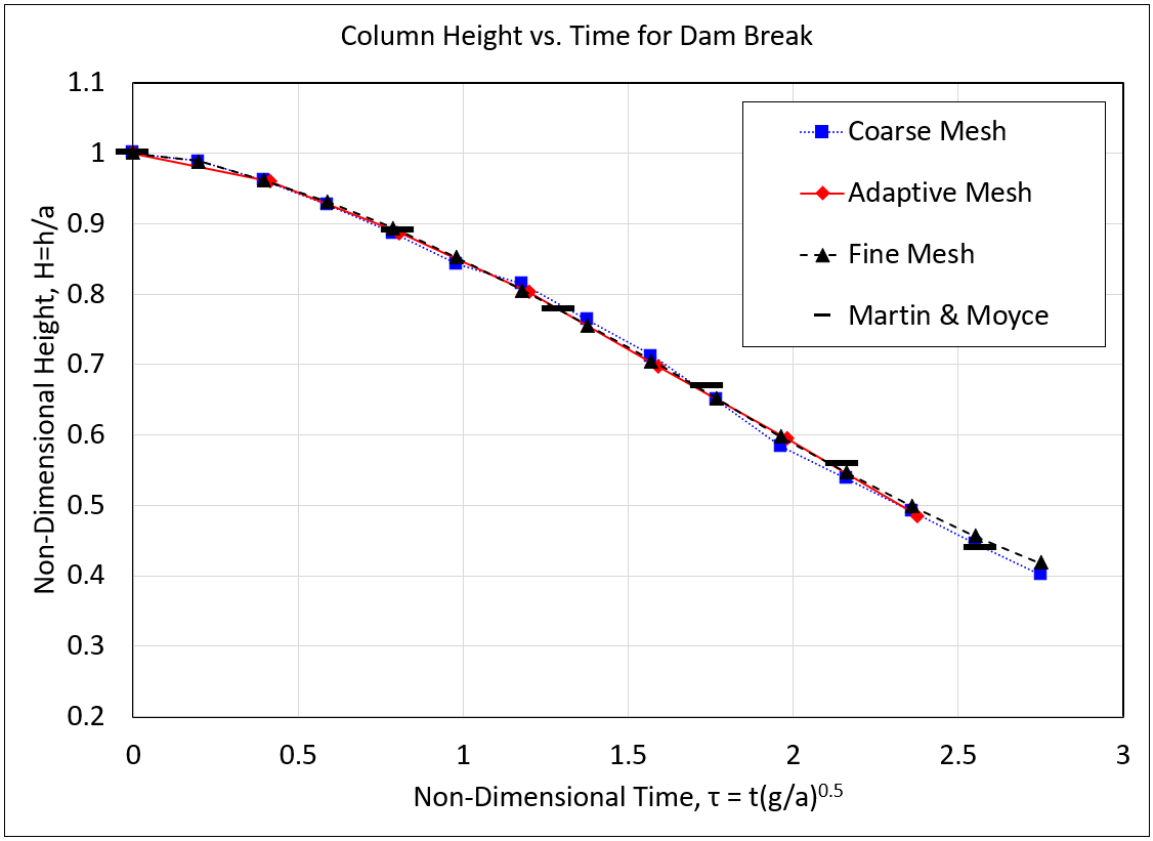


Figure 5.7: The non-dimensional column heights are shown plotted versus non-dimensional time for each of the three mesh variations for redistancing with a global volume constraint and for Martin and Moyce's experimental results.

Averaging the adaptation time over the entire solution gives and average runtime per time step of 0.914 s, compared to the uniform fine mesh runtime per time step of 2.3 s.

From these cases it is clear that mesh refinement is an effective means of improving volume conservation, and that volume conservation behavior is more sensitive to mesh size than it is to redistancing variation. It is further clear that eliminating the volume change during redistancing (i.e., allowing no interface motion during redistancing) and minimizing the volume change during redistancing (through the application of a global volume constraint) do not result in fully conservative methods regardless of mesh size or adaptivity. This is because there are sources of volume change in addition to interface motion during redistancing.

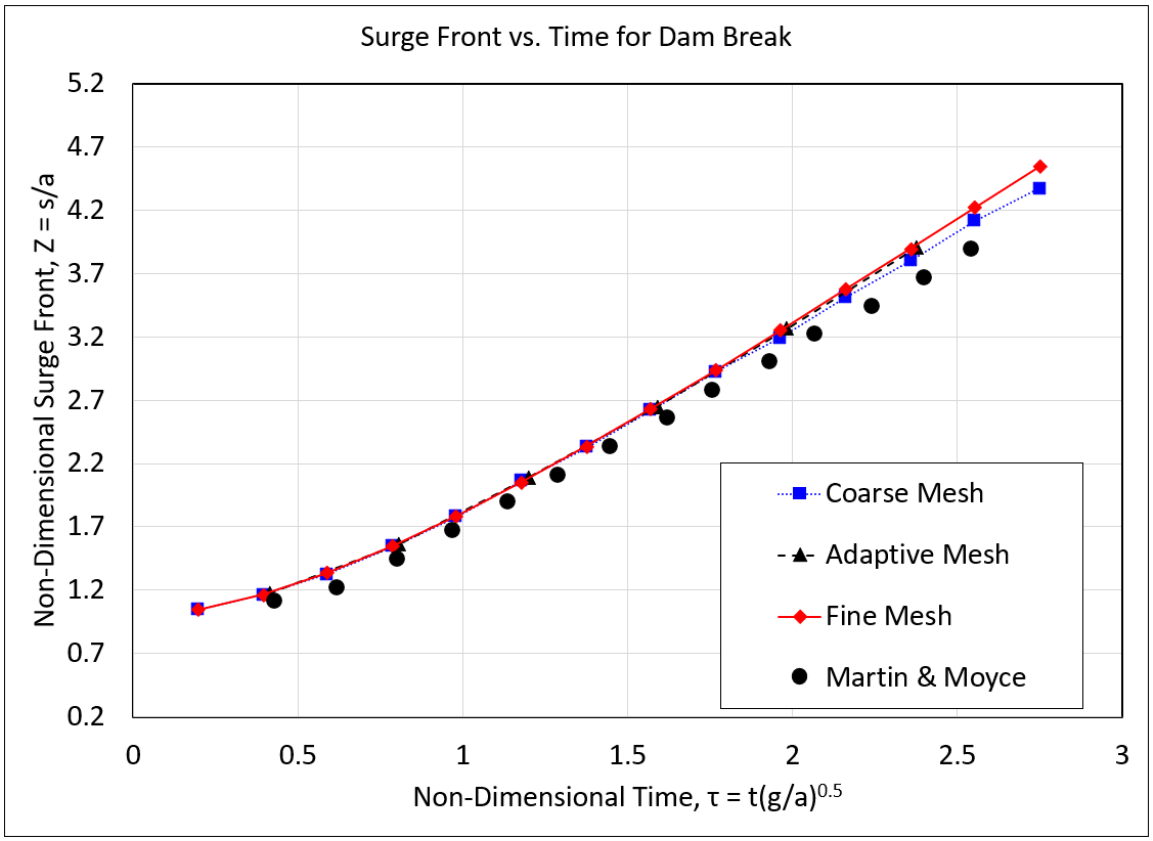


Figure 5.8: The non-dimensional surge front lengths are shown plotted versus non-dimensional time for each of the three mesh variations for redistancing with a global volume constraint and for Martin and Moyce's experimental results.

Prior to this research it was clear that the previously available options of unconstrained redistancing and redistancing with a global volume constraint were prone to significant motion of the interface during the redistancing, especially on coarse grids or grids that marginally resolved the interface (e.g., regions of high curvature). Indeed this was the motivation for the development of CLSVOF (unsuccessful) and allowing no interface motion during redistancing (pinning the interface position during the redistancing operation). However, while the redistancing with no interface motion was validated to yield no volume change during the redistancing step, it did not always result in the substantial improvements to volume conservation that might have been expected. The reason for this limited improvement requires consideration of the rest of the flow solution algorithm. Specifically,

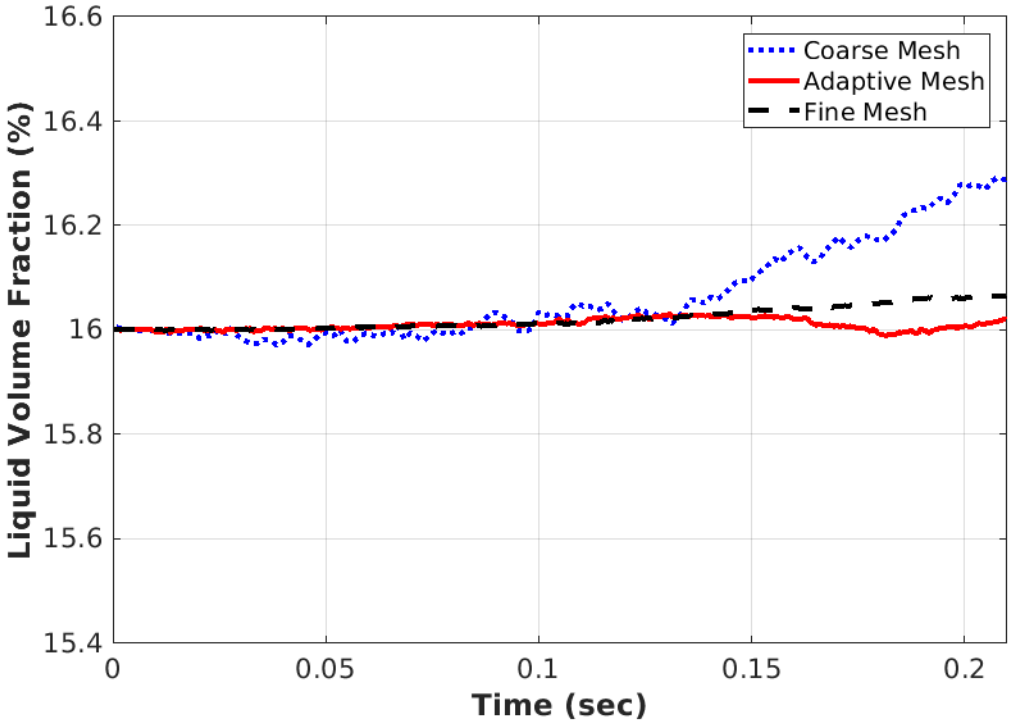


Figure 5.9: The history of the volume fraction for each mesh over the course of a 0.21 s simulation with redistancing with a global volume constraint. The coarse mesh is shown in blue, the adaptive mesh in red, and the uniform fine mesh in black.

prior to redistancing, on every time step, there are multiple iterations of the flow and level set scalar (alternating) solution steps. While these flow and scalar solution steps are globally conservative, they are not locally conservative and it is apparent from the results of the studies of redistancing with no interface motion that there is substantial volume change on coarse meshes (or coarsely resolved features). It is further clear, that local conservation improves dramatically when the interface is better resolved either by fine uniform meshes or by adaptive meshes.

There are two somewhat more subtle points that need to be reiterated and made more clear. First, the pinning of the interface in the algorithm for redistancing with no interface motion, while fully conservative in terms of the redistancing operation, is not without negative consequences. Because all of the nodes of the elements cut by the interface are

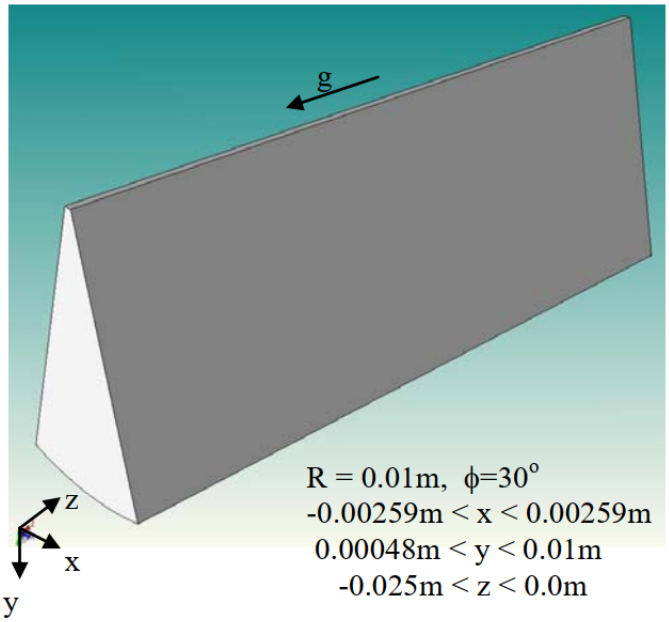
unchanged, their values are allowed to be incorrect, that is, their values are not necessarily true distances to the interface. The reason for this is simple: the redistancing algorithm starts from a level set field that was advected by the spatially varying velocity, and the spatial variation introduces distortion of the distance field. For the pinning operation to be successful (i.e., result in no interface motion), it must use the advected value of the level set, even if that value is smaller (i.e., shows that a node is closer to the interface than it actually is) or larger than it should be for a true distance field. If these values are corrected, that correction process itself moves the interface (because the interface is defined using the level set values at the surrounding nodes, changing the level set value at one or more of those nodes moves the interface), and the primary goal of the algorithm would be lost – redistancing would result in a volume change. It is not clear at this time (subject of future research) how big an impact this local error to the distance field due to elements being pinned to an incorrect distance has on the flow solution and on mass conservation. It is certain that there is some interaction since during the next flow solution step, the viscosity and density are blended using a distance field that is distorted within the blending distance because these pinned values are by definition a fraction of an element size/length away from the interface, and the blending distance is typically two to four elements.

Again, mesh refinement improves the situation. That said, adaptive mesh refinement introduces an additional source of conservation error – solution interpolation. When the mesh is adapted, element edges are split in regions requesting sizes smaller than the current mesh at that location and collapsed in regions requesting larger sizes than the current mesh in that location. For a refinement, the newly inserted vertex (e.g., on the mid-point of the edge) gets an interpolation of the solution from the current mesh (average in the case of a single split at the mid point). While this preserves the order of accuracy it is not strictly conservative. A similar situation occurs during coarsening, where an edge collapse would result in the solution from the edge endpoints on the original mesh being averaged and given to the adapted mesh – again accuracy is preserved but, as with refinement, conservation is

not guaranteed. While it is possible to develop solution transfer algorithms that are fully conservative, they are substantially more expensive. In this research, it appears that, so long as adaptivity maintains a fine grid in regions of high curvature interface, the solution transfer (interpolation) error is of the same order as the other errors making the current approach practical and efficient.

5.3 Annular Flow

The annular flow simulations are based on the work of Rodriquez [65, 66]. The computational domain, shown in Figure 5.10, is a 30 degree wedge of a 0.02 m diameter tube. The domain is 0.025 m long. Periodicity is enforced in the axial and circumferential directions. The model is clipped at a radius of 0.0005 m to prevent singularity issues along the centerline. An effective-viscosity turbulent wall function is used along the tube wall, where the size of the mesh along the wall is small enough to ensure the first off-the-wall nodes are within the viscous sublayer to prevent the need to know the log-law region coefficients.



Boundary Conditions
x: periodic (circumferential)
y: wall & symmetry
z: periodic (axial)

Initial Conditions:
Flow: Phasic plug flow
Level Set: δ_m (with perturbation)

Imposed experimental pressure gradient.

Properties

	Water	Steam
ρ (kg/m^3)	740	36.53
μ: (Pa-s)	9.13E-5	1.90E-5
σ: (N/m)	0.0177	

Figure 5.10: The problem domain for annular flow simulations. Image modified from one created by J. Rodriquez [65]; used with permission.

The initial mesh for the annular flow simulations is an isotropic tetrahedral mesh that is fine at the wall (element size of $1.0*10^{-5}$ m) and grades to the non-wall resolution (element size of $2.0*10^{-4}$ m) with a gradation factor of 0.3. The mesh has a total of approximately 32M elements. The initial annular flow mesh is shown in Figure 5.11. The adaptive refinement case A2 is created from a flow solution on the initial mesh after five time steps. Subsequent adaptive cases are derived from case A2.

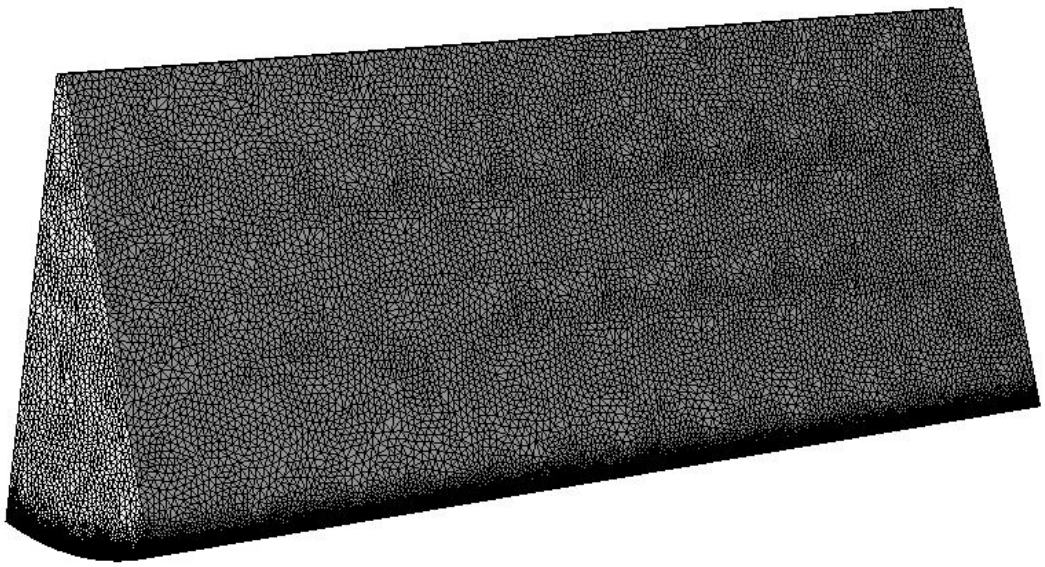


Figure 5.11: The initial mesh for annular flow simulations.

In the adaptive refinement cases, there are four adaptation bands, as listed in Table 5.2. The first adapt size, $2.0 * 10^{-4}$ m, is the same as the non-wall resolution of the initial mesh. The corresponding band size is 0.00 m for all cases, meaning that the band is inactive. It is included, though, for the possibility of controlling coarsening to the original mesh size. The second adapt size creates elements that are of size $6.0 * 10^{-5}$ m and within $0.70e * 10^{-3}$ m of the interface. It is active in cases A2 and A2-C1. The third adapt size is finer at $2.5*10^{-5}$ m, and is used in case A2 within $1.5 * 10^{-4}$ m of the interface and in case A2-C2 over the entire

domain. The final adapt size, $1.0*10^{-5}$ m, is not used as a band about the interface; instead, it is used in all cases to ensure that the mesh at the wall stays sufficiently fine.

Adapt Size	$2.0 * 10^{-4}$	$6.0 * 10^{-5}$	$2.5 * 10^{-5}$	$1.0 * 10^{-5}$
Band Size - A2	0.00	$0.70e * 10^{-3}$	$1.5 * 10^{-4}$	0.00998
Band Size - A2-C1	0.00	$0.70e * 10^{-3}$	0.00	0.00998
Band Size - A2-C2	0.00	0.00	whole domain	0.00998

Table 5.2: The adaptive element and band sizes for the annular flow adaptation cases. All numbers are given in meters.

In adaptive refinement case A2 there are two refined mesh bands around the interface, as discussed above. The adaptive meshes for this case have approximately 171M elements. An example mesh is shown in Figure 5.12. To ensure that the interface remains within the narrow, more refined band, the solve/adapt cycle has only 20 time steps of flow solution between mesh adaptations. An example of the interface for the A2 case is shown in Figure 5.13. This interface shows developed flow with breaking waves, droplets, bubbles, and fluid ligaments.

Two additional adaptive cases were developed from time step 3455 of case A2. Step 3455 was chosen because it has well developed flow with bubbles, drops, and waves. Creating the two additional cases from case A2 enables an accurate comparison of volume conservation because all three cases (considering case A2 to start at step 3455 for the purposes of volume analysis) have the same history and therefore the same starting volume fraction and flow features. Case A2 was run for an additional 2000 time steps (including mesh adaptations every 20 time steps) from the point were the additional cases were created.

In the first case created from case A2, case A2-C1, the inner, narrower refinement band is coarsened away so that there is only the broader refinement band, as described in Table 5.2. The meshes for this case have approximately 83M elements. An example mesh

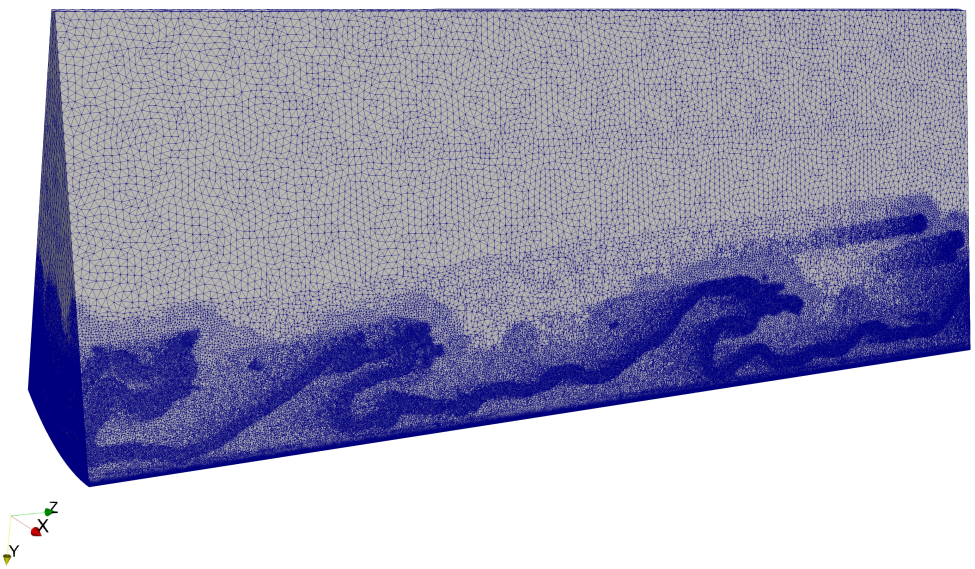


Figure 5.12: An example mesh for annular flow case A2, showing both the narrow and broad refinement bands.

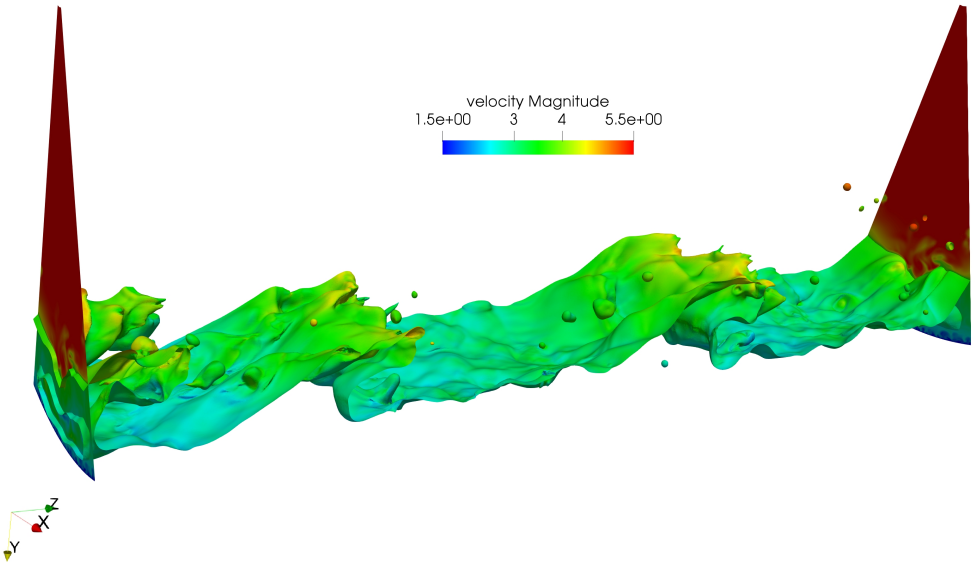


Figure 5.13: An example interface for annular flow case A2, showing well developed flow with waves, droplets, and bubbles.

is shown in Figure 5.14. Because the interface refinement band in this case is broad, there

are 150 time steps between mesh adaptations. An example of the interface for the A2-C1 case is shown in Figure 5.15. This interface shows the same types of flow features as the A2 interface, but there are fewer droplets and bubbles, and the curvature of all features is lower.

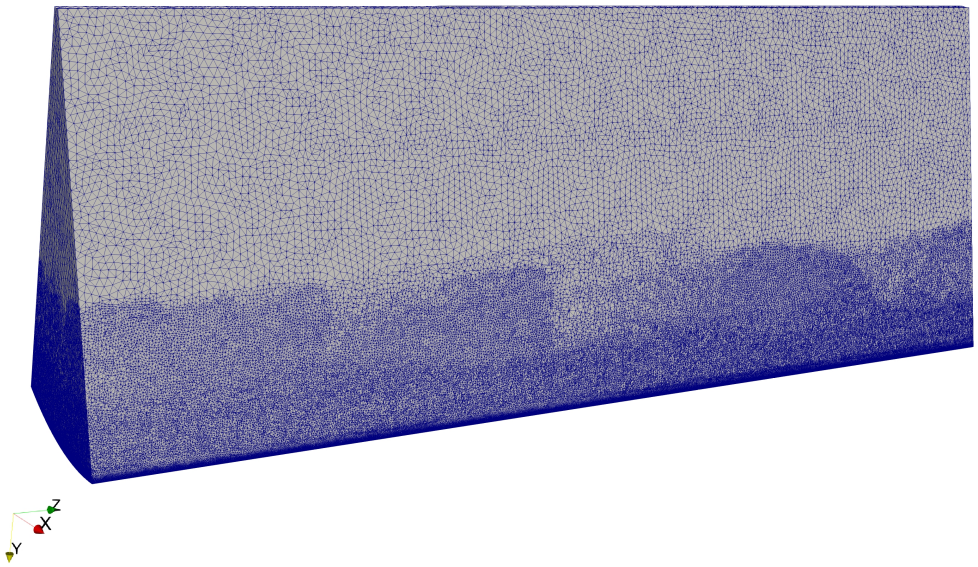


Figure 5.14: An example mesh for annular flow case A2-C1, showing the broad refinement band.

In the second adaptive case created from case A2, case A2-C2, the inner refinement band is expanded to cover the entire computational domain and approximate a fine uniform mesh with a finer wall mesh (as described in Table 5.2). The mesh for case A2-C2 consists of approximately 563M elements. Because the mesh is approximately uniform, case A2-C2 is not adapted further. Figure 5.16 shows the mesh for case A2-C2, and Figure 5.17 shows an example interface from the A2-C2 simulation. Compared to case A2-C1, case A2-C2 has more fine bubbles, droplets, and fluid ligaments. These are areas of high curvature that the coarser mesh of A2-C1 is unable to resolve. Note, however, that A2 (Figure 5.13) and A2-C2 (Figure 5.17) show similar interfacial features owing to the fact that A2 preserves the same resolution of the interface as A2-C2 through local adaptivity in a narrow band around the interface.

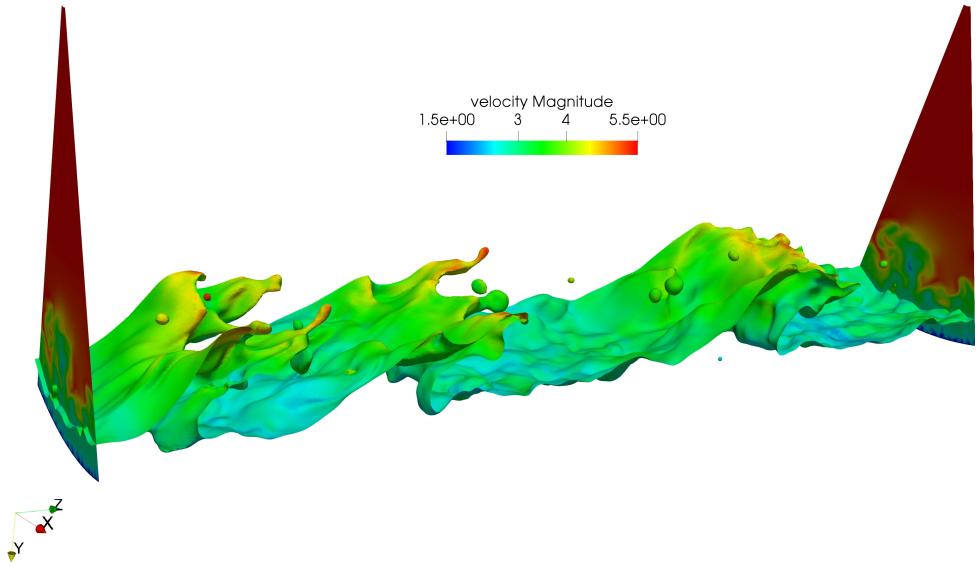


Figure 5.15: An example interface for annular flow case A2-C1, showing well developed flow with waves, droplets, and bubbles.

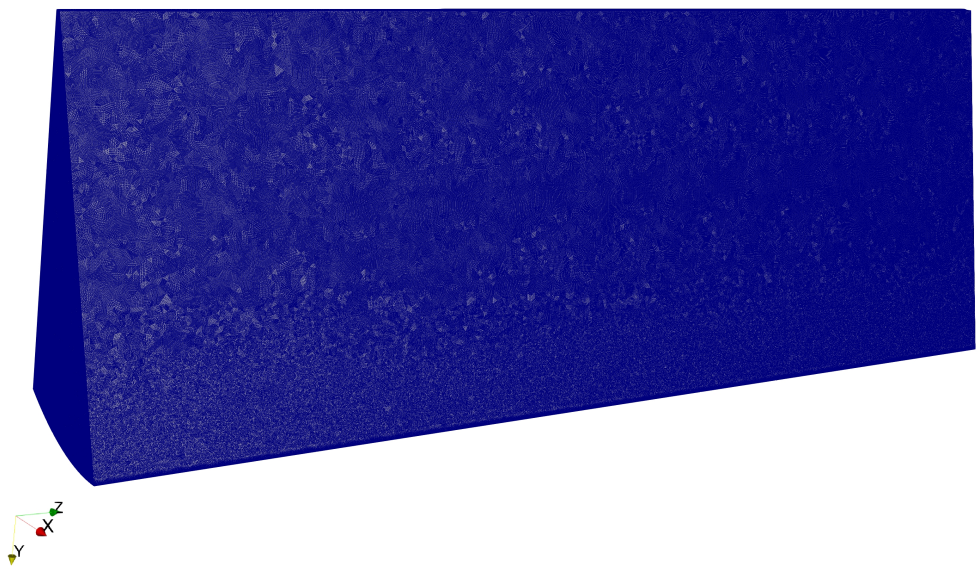


Figure 5.16: The mesh for annular flow case A2-C2.

Figure 5.18 shows the volume fraction histories for the three annular flow cases. In the history for case A2, there is a steady loss of liquid volume both before and after the

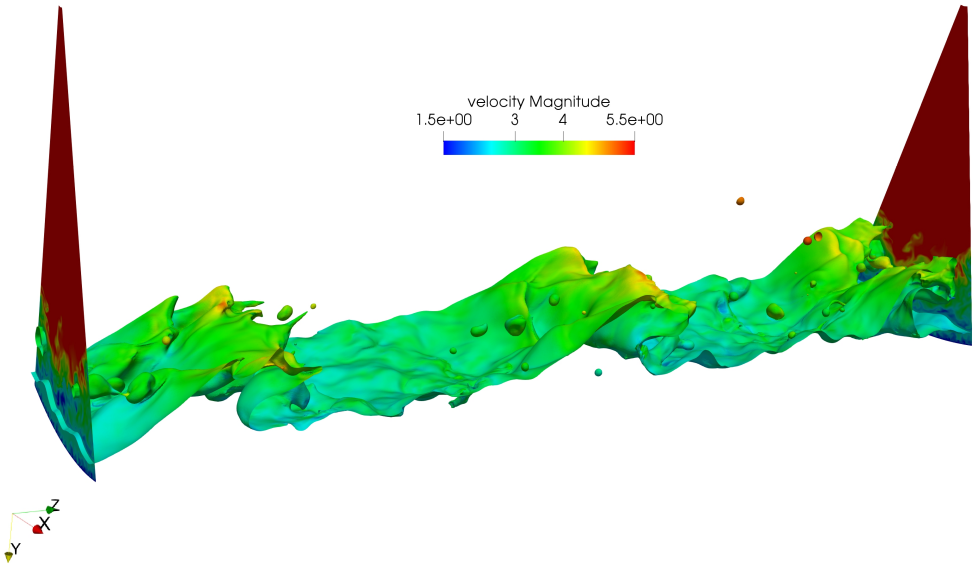


Figure 5.17: An example interface for annular flow case A2-C2, showing well developed flow with waves, droplets, and bubbles.

creation point for cases A2-C1 and A2-C2. The volume fraction history for case A2-C2 very closely follows that of case A2, which is expected since the two cases similarly resolve the interface. However, case A2-C1, which resolves the interface less well, has better global volume conservation. This is likely due to the loss of high curvature regions (droplets, bubbles, and ligaments) fortuitously canceling each other out so that although volume is not conserved locally, it is conserved globally. This is possible because the main source of volume loss is the loss of high curvature regions, and the phase that is on the concave side of the interface is the one to experience local volume loss.

The continued loss of volume fraction for cases A2 and A2-C2 indicates that there are more fine structures in the interface that are concave toward the liquid phase and that the high curvature of these structures is not fully resolvable, leading to volume loss of the liquid. This indicates that even with the fine mesh, not all scales of the interface are being resolved; that is, this is not a direct numerical simulation. It is clear that a sub-grid scale phase model is needed in order to more accurately track the interface in a cost-efficient manner. The

alternative of refining the interface until even the smallest ligaments and drops (or sprays) are resolved would be orders of magnitude more expensive.

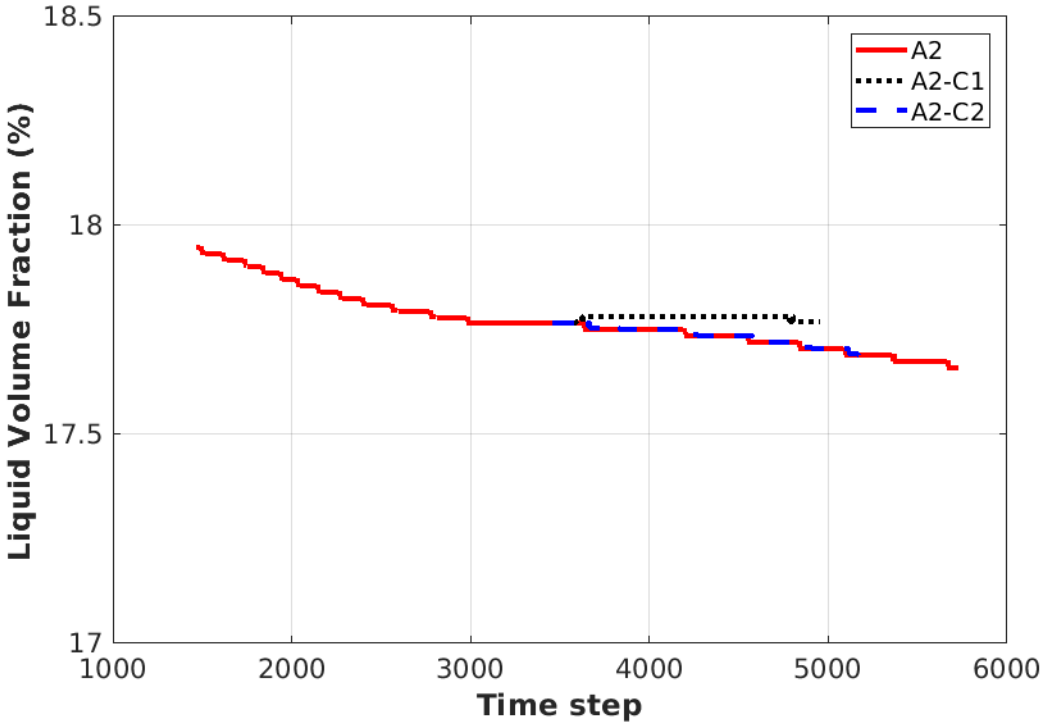


Figure 5.18: Volume fraction histories for the annular flow cases. Case A2 is shown in red. Case A2-C1 (the coarse refinement band) is shown with a dotted black line. Case A2-C2 (the uniform fine mesh) is shown with a dashed blue line.

Each of these adaptive cases comes with a different computational cost. Cases A2 and A2-C1 have the cost of periodic mesh adaptation, in addition to the cost of flow solution. The three cases also have very different problem sizes, leading to different computational costs for the flow solution. Table 5.3 summarizes the mesh statistics, computational costs, and solver efficiency for each case. Each of the annular flow cases was run on Blue Gene/Q resources at Argonne National Laboratory; these resources can accommodate one, two, or four processes per computing core, and each computing node has sixteen cores. Running two or four processes per core does not increase the computational resource required. Cases A2 and A2-C1 are run using two processes per core, and case A2-C2 is run using four processes

per core. The runtimes listed are average wall-clock time values. The compute costs are calculated by multiplying the runtimes by the number of computing nodes used.

As the data in Table 5.3 show, case A2-C2 has the greatest solver efficiency (smallest compute time per time step per million vertices) and the lowest computational cost. The mesh partitioning for case A2-C2 is well balanced to enable PHASTA to run at its most efficient. Furthermore, A2-C2 gains further efficiency by running 4 processes per core which has been shown to give an additional 30% speedup. For cases A2 and A2-C1, there are two trade-offs to be made that affect efficiency. The first trade-off is the frequency of adaptation versus the size of the adaptation bands. Larger bands allow for less frequent adaptation but increase the computational cost of the flow solution because the mesh is larger. The second trade-off is in part balancing in each adaptation cycle. Better balanced parts enable PHASTA to run more efficiently and therefore reduce the computational cost of the flow solution, but part balancing can be a computationally expensive part of the adapt cycle. In both cases A2 and A2-C1, the adapt cycle is set to accept imperfectly balanced parts in order to reduce the adaptation time, but this results in poorer solver efficiency for the flow solution. Neither trade-off balance has been shown to be optimal for either case A2 or A2-C1. Regardless of the optimization of the solve/adapt cycles, it is clear that until the adaptation can be made much less computationally expensive, for large problems with complex interfaces, it is less expensive to solve the flow on a fine uniform mesh than it is to use an adaptive mesh scheme.

While A2 failed to win in terms of computational time it is important to note that it did provide an equally accurate solution. A secondary metric of a simulation is the size of the data that it creates. In this metric A2 is a clear winner since its solution files are more than eight times smaller. A third metric to consider is the size of the computational resource required. A2 was able to run on smaller resources than A2-C2, though this gain was mostly offset by the large memory requirements of the adaptation process.

	Case A2	Case A2-C1	Case A2-C2
Number of mesh vertices	30,232,072	15,166,147	131,473,571
Number of elements	171,192,070	82,705,039	562,836,441
Partition count	32,768	32,768	131,072
Number of computing nodes	16k	16k	32k
Runtime per time step	20.63 s	12.16 s	23.87 s
Compute cost per time step	330.08 ks	194.56 ks	763.64 ks
Compute cost per time step per million vertices	10.92 s	12.83 s	5.81 s
Runtime per adapt cycle	1306.24 s	1965.98 s	–
Adapt frequency	20	150	–
Adapt time per time step	65.31 s	13.11 s	–
Adapt compute cost per time step	1044.99 ks	209.76 ks	–

Table 5.3: Mesh and solver statistics for the annular flow cases. The runtime per time step and the runtime per adapt cycle are average wall-clock time values.

5.4 Summary

Mesh refinement is an effective way to improve volume conservation. The variation in volume conservation is greater between meshes of different sizes than it is between different redistancing methods on the mesh.

Global volume conservation does not tell the whole story because the loss of high curvature regions can cancel each other out and have no global effect, meaning that a simulation may have improved volume conservation while simultaneously producing a less accurate interface.

Adaptive mesh refinement can offer reduced computational cost, as compared to uniform refinement, but it does not necessarily do so. It depends on how expensive the adapt cycles are and how frequently the mesh is adapted. For example, AMR offered a factor of two improvement in computational cost over uniform refinement for the dam break problem, but AMR was a factor of two more expensive than uniform refinement for the annular flow problem.

Chapter 6

Conclusions

6.1 Conclusions

The goal of this work was to improve interface tracking performance for incompressible two-phase flows on parallel unstructured meshes. This goal was pursued first through the effort to develop coupled level set volume of fluid (CLSVOF) capability for parallel unstructured grids (CLSVOF had been previously demonstrated on serial, structured grids). When the CLSVOF development was unsuccessful, the goal of improving interface tracking performance was pursued through the development of improved understanding of level set options and trade-offs, including the use of different level set redistancing schemes and mesh refinement options.

The effort to develop CLSVOF for parallel unstructured grids was unsuccessful. There are two main contributing factors to the lack of success. The first is that CLSVOF depends on having a non-diffusive method for advecting the volume fraction. The simple upwinding scheme used here for advecting the volume fraction was too numerically diffusive to enable accurate reconstruction of the interface. The second factor is that CLSVOF also depends on having a reliable interface reconstruction algorithm that both does not move the interface and that re-establishes a true distance field. Developing a suitable interface reconstruction algorithm was a challenge in this work that was not overcome, due in part to the elemental interdependence of level set values (i.e., changing the level set value at one node based on

the interface location in one of its adjacent elements affects the interface location for any other adjacent elements also containing the interface).

Studies of level set redistancing algorithms and meshing strategies provided an alternate means of potentially improving interface tracking performance. This work looked at three redistancing algorithms - redistancing with no constraints, redistancing with a global volume constraint, and redistancing with no interface motion - and two mesh refinement approaches - uniform refinement and interface-driven adaptive refinement. The redistancing algorithms were evaluated for their computational cost, accuracy, ease of implementation, parallelizability, volume conservation, and preservation of curvature. No single redistancing variation had the best performance in every category. Unconstrained redistancing was the most easily implemented, most parallelizable, and least computationally expensive redistancing variation. Based on the canonical test problems of Chapter 4, redistancing with a global volume constraint performs better (in terms of accuracy, preservation of curvature, and volume conservation) than unconstrained redistancing, but at a significant increase in computational expense. Redistancing with no interface motion offers reduced computational expense and improved preservation of curvature relative to redistancing with a global volume constraint. The mesh refinement studies of Chapter 5 revealed an additional performance measure for the redistancing algorithms - robustness. The unconstrained redistancing and redistancing with no interface motion both exhibited poor performance on refined and adaptive meshes that is not yet fully understood. The mesh refinement studies nonetheless showed that mesh refinement is an effective means of improving the accuracy, volume conservation, and preservation of curvature for a robust algorithm. Whether a uniformly refined mesh or an adaptive meshing strategy is less computationally expensive depends on both the meshing strategy and the solver performance for a given question. Overall, the best means of improving interface tracking for the level set is to use a global volume constraint during redistancing and conduct the problem on a mesh that is fine near the interface (if not everywhere) to minimize loss of high curvature regions.

Many level set development efforts (and related efforts such as CLSVOF) have focused on improving volume conservation as their main goal. However, the close study of redistancing algorithms showed the extent to which volume conservation is limited as a performance measure. For example, global volume conservation is usually considered independently of local volume conservation. This means that an algorithm may have very good global volume conservation but also very poor preservation of curvature, as high curvature regions are lost and volume is numerically redistributed (in a non-physical way) to lower curvature regions of the domain. Improving volume conservation and minimizing interface motion have generally been considered synonymous. However, a redistancing method may be very good at not allowing interface motion but still lead to poor overall volume conservation because the method also preserves large gradients in the level set close to the interface.

Additionally, volume changes during level set redistancing are only one source of poor volume conservation - flow solution and advection of the level set can also produce volume changes. These changes can be reduced by better resolving the flow or by improving the accuracy of the level set advection scheme. Spatial discretization is also a source of local volume changes, and therefore possible global volume changes. Any mesh is limited in its representation of small interface features (i.e., droplets, bubbles, and fluid ligaments). These high curvature features will all tend to locally lose volume for whichever fluid is on the concave side of the interface (e.g., droplets will shrink and lose liquid volume), unless they are large enough that the mesh can fully represent their curvature. This local loss of volume is clearly inaccurate, but may not be apparent in consideration of global volume conservation because local changes in volume may offset each other. This effect of spatial discretization can be mitigated by higher resolution meshes, or potentially by higher order representation of the level set or a sub-grid scale phase model.

6.2 Recommendations for Future Work

It may be possible to develop an effective CLSVOF algorithm for unstructured parallel meshes, and such an algorithm would be potentially beneficial to the simulation of complex flows such as annular flow. In order to have an effective CLSVOF algorithm within PHASTA, there are two needs. The first is an implementation of a non-diffusive volume fraction advection algorithm. Such algorithms have been demonstrated on unstructured meshes (see, for example, Zhao, et al. [96]), but none have been implemented in PHASTA. The second need is the development of an interface reconstruction algorithm that handles the elemental interdependence of the level set. It may be possible to address both needs in a single algorithm, such as a semi-Lagrangian scheme that advances the volume fraction and interface and then projects them back to the Eulerian mesh that the flow is solved on.

For two-phase flows, the area for which the greatest mesh refinement is desired is the area around the interface. Finer meshes enable the resolution of finer interface features and therefore more accurate simulations. It is apparent from this work that there are still improvements that can be made in the computational expense of adaptive meshing, and that these improvements must be made to make adaptive mesh refinement a viable tool for all large, complex flow simulations.

Finally, this work showed that the unconstrained redistancing and redistancing without interface motion level set variations are insufficiently robust to be used on a wide range of problems and meshes. Their poor behavior may be related to under-resolved flow solutions, or to sensitivity to interface thicknesses, or to some other cause affecting them but not redistancing with a global volume constraint. Whatever the cause, it should be diagnosed and corrected, if possible. If the cause cannot be corrected, then a guide for working with those level set variations should be established (e.g., they require a smaller or larger interface thickness than the global volume constraint variation), so that the use of these variations on more complex problems is a viable option. It may be that if the robustness problem can

be corrected, one of those two variations on a very fine mesh may offer a better balance of accuracy, volume conservation, and computational expense than redistancing with a global volume constraint.

Bibliography

[1] H. T. Ahn and M. Shashkov. Multi-material interface reconstruction on generalized polyhedral meshes. Journal of Computational Physics, 226(2):2096–2132, 2007.

[2] H. T. Ahn and M. Shashkov. Adaptive moment of fluid method. Journal of Computational Physics, 228:2792–2821, 2009.

[3] H. T. Ahn, M. Shashkov, and M. A. Christon. The moment-of-fluid method in action. Communications in Numerical Methods in Engineering, 25:1009–1018, 2009.

[4] J. E. Akin, T. E. Tezduyar, and M. Ungor. Computation of flow problems with the mixed interface-tracking/interface-capturing technique (mitict). Computers & Fluids, 36(1):2–11, 2007.

[5] I. Akkerman, Y. Bazilevs, D. Benson, M. Farthing, and C. Kees. Free-surface flow and fluid-object interaction modeling with emphasis on ship hydrodynamics. Journal of Applied Mechanics, 79(1):010905, 2012.

[6] R. Ardon and L. D. Cohen. Fast constrained surface extraction by minimal paths. International Journal of Computer Vision, 69(1):127–136, 2006.

[7] I. A. Bolotnov. Influence of bubbles on the turbulence anisotropy. Journal of Fluids Engineering, 135, 2013.

[8] I. A. Bolotnov, K. E. Jansen, D. A. Drew, A. A. Oberai, R. T. L. Jr., and M. Z. Podowski. Detached direct numerical simulation of turbulent two-phase bubbly channel flow. International Journal of Multiphase Flow, 37:647–659, 2011.

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

[10] Z. Cao, D. Sun, J. Wei, and B. Yu. A coupled volume-of-fluid and level set method based on multi-dimensional advection for unstructured triangular meshes. Chemical Engineering Science, 176:560–579, 2018.

[11] Z. Cao, D. Sun, B. Yu, and J. Wei. A coupled volume-of-fluid and level set (voset) method based on remapping algorithm for unstructured triangular grids. International Journal of Heat and Mass Transfer, 111:232–245, 2017.

[12] I. Chakraborty, G. Biswas, and P. S. Ghoshdastidar. A coupled level-set and volume-of-fluid method for the buoyant rise of gas bubbles in liquids. International Journal of Heat and Mass Transfer, 58:240–259, 2013.

[13] I. Chakraborty, M. Rubio-Rubio, A. Sevilla, and J. Gordillo. Numerical simulation of axisymmetric drop formation using a coupled level set and volume of fluid method. International Journal of Multiphase Flow, 84:54–65, 2016.

[14] Y. C. Chang, T. Y. Hou, B. Merriman, and S. Osher. Eulerian capturing methods based on a level set formulation for incompressible fluid interfaces. Journal of Computational Physics, 124:449, 1996.

[15] A. Cohen, N. Dyn, and D. Levin. Matrix subdivision schemes. Technical report, Université Pierre et Marie Curie, 1997.

[16] M. A. Cruchaga, D. J. Celentano, and T. E. Tezduyar. A numerical model based on the mixed interface-tracking/interface-capturing technique (mitict) for flows with fluid-solid and fluid-fluid interfaces. International journal for numerical methods in fluids, 54(6-8):1021–1030, 2007.

[17] V. Dyadechko and M. Shashkov. Moment of fluid interface reconstruction. Technical Report LA-UR-05-7571, Los Alamos National Laboratory, 2007.

[18] V. Dyadechko and M. Shashkov. Reconstruction of multi-material interfaces from moment data. Journal of Computational Physics, 227:5361–5384, 2008.

[19] L. B. Fore, S. G. Beus, and R. C. Bauer. Interfacial friction in gas-liquid annular flow: analogies to full and transition roughness. International Journal of Multiphase Flow, 26(11):1755–1769, 2000.

[20] L. P. Franca and S. Frey. Stabilized finite element methods: II. The incompressible Navier-Stokes equations. Comp. Meth. Appl. Mech. Engng., 99:209–233, 1992.

[21] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape priors. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 1, pages 755–762. IEEE, 2005.

[22] T. Fukano and T. Inatomi. Analysis of liquid film formation in a horizontal annular flow by dns. International Journal of Multiphase Flow, 29(9):1413–1430, 2003.

[23] M. Fulgosi, D. Lakehal, S. Banerjee, and V. De Angelis. Direct numerical simulation of turbulence in a sheared air–water flow with a deformable interface. Journal of fluid mechanics, 482:319–345, 2003.

[24] C. Gear. Numerical initial value problems in ordinary differential equations. Prentice-Hall, Englewood Cliffs, NJ, 1971.

[25] D. Gueyffier, J. Li, A. Nadim, R. Scardovelli, and S. Zaleski. Volume-of-fluid interace tracking with smoothed surface stress methods for three-dimensional flows. Journal of Computational Physics, 152:423–456, 1999.

[26] W. Henstock and T. Hanratty. The interfacial drag and the height of the wall layer in annular flows. AIChE Journal, 22(6):990–1000, 1976.

[27] G. F. Hewitt and N. S. Hall-Taylor. Annular two-phase flow. Pergamon Press, 1970.

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

[29] A. Hoogi, C. F. Beaulieu, G. M. Cunha, E. Heba, C. B. Sirlin, S. Napel, and D. L. Rubin. Adaptive local window for level set segmentation of ct and mri liver lesions. Medical image analysis, 37:46–55, 2017.

[30] H. H. Hu. Direct simulation of flows of solid-liquid mixtures. International Journal of Multiphase Flow, 22(2):335–352, 1996.

[31] H. H. Hu, N. A. Patankar, and M. Zhu. Direct numerical simulations of fluid–solid systems using the arbitrary lagrangian–eulerian technique. Journal of Computational Physics, 169(2):427–462, 2001.

[32] K. E. Jansen. http://www.scorec.rpi.edu/ kjansen/phasta. CFD code.

[33] K. E. Jansen, S. S. Collis, C. H. Whiting, and F. Shakib. A better consistency for low-order stabilized finite element methods. Comp. Meth. Appl. Mech. Engng., 174:153–170, 1999.

[34] K. E. Jansen and M. Rasquin. Petascale, adaptive CFD (ALCF ESP technical report): ALCF-2 early science program technical report. Technical Report ANL/ALCF/ESP-13/7, Argonne National Laboratory, 2013.

[35] K. E. Jansen, C. H. Whiting, and G. M. Hulbert. A generalized-α method for integrating the filtered Navier-Stokes equations with a stabilized finite element method. Comp. Meth. Appl. Mech. Engng., 190:305–319, 2000.

[36] M. Jemison, E. Loch, M. Sussman, M. Shashkov, M. Arienti, M. Ohta, and Y. Wang. A coupled level set-moment of fluid method for incompressible two-phase flows. Journal of Scientific Computing, 54(2-3):454–491, 2013.

[37] M. Jemison, M. Sussman, and M. Arienti. Compressible, multiphase semi-implicit method with moment of fluid interface representation. Journal of Computational Physics, 279:182–217, 2014.

[38] M. Jemison, M. Sussman, and M. Shashkov. Filament capturing with the multimaterial moment-of-fluid method. Journal of Computational Physics, 285:149–172, 2015.

[39] R. Kashyap and P. Gautam. Fast level set method for segmentation of medical images. In Proceedings of the International Conference on Informatics and Analytics, page 20. ACM, 2016.

[40] I. Kataoka, M. Ishii, and K. Mishima. Generation and size distribution of droplet in annular two-phase flow. J. Fluids Eng, 105:230–238, 1983.

[41] R. Kumar, C. C. Maneri, and T. D. Strayer. Modeling and numerical prediction of flow boiling in a thin geometry. Journal of Heath Transfer-Transactions of the ASME, 126(1):22–33, 2004.

[42] R. Kurose, N. Takagaki, A. Kimura, and S. Komori. Direct numerical simulation of turbulent heat transfer across a sheared wind-driven gas–liquid interface. Journal of Fluid Mechanics, 804:646–687, 2016.

[43] M. Kwakkel, W.-P. Breugem, and B. J. Boersma. Extension of a CLSVOF method for droplet-laden flows with a coalescence/breakup model. Journal of Computational Physics, 253:166–188, 2013.

[44] R. Lahey and D. Drew. The analysis of two-phase flow and heat transfer using a multidimensional, four field, two-fluid model. Nuclear Engineering and Design, 204(1-3):29–44, 2001.

[45] R. T. Lahey. The simulation of multidimensional multiphase flows. Nuclear Engineering and Design, 235(10-12):1043–1060, 2005.

[46] D. Lakehal, M. Meier, and M. Fulgosi. Interface tracking towards the direct simulation of heat and mass transfer in multiphase flows. International Journal of Heat and Fluid Flow, 23(3):242–257, 2002.

[47] C. Li, R. Huang, Z. Ding, J. Gatenby, D. N. Metaxas, J. C. Gore, et al. A level set method for image segmentation in the presence of intensity inhomogeneities with application to mri. IEEE Transactions on Image Processing, 20(7):2007, 2011.

[48] X. Li, M. Shephard, and M. Beall. 3D anisotropic mesh adaptation by mesh modification. Comp. Meth. Appl. Mech. Engng., 194(48-49):4915–4950, 2005.

[49] J. Lopez, J. Hernandez, P. Gomez, and F. Faura. An improved plic-vof method for tracking thin fluid structures in incompressible two-phase flows. Journal of Computational Physics, 208(1):51–74, 2005.

[50] J. C. Martin and W. J. Moyce. Part iv. an experimental study of the collapse of liquid columns on a rigid horizontal plane. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 244(882):312–324, 1952.

[51] MATLAB. version 8.3.0.532 (R2014a). The MathWorks Inc., Natick, Massachusetts, 2014.

[52] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. Communications on pure and applied mathematics, 42(5):577–685, 1989.

[53] M. Muradoglu and G. Tryggvason. A front-tracking method for computation of interfacial flows with soluble surfactants. Journal of computational physics, 227(4):2238–2262, 2008.

[54] S. Nagrath. Adaptive stabilized finite element analysis of multi-phase flows using level set approach. MANE, Rensselaer Polytechnic Institute, 130, 2004.

[55] S. Nagrath, K. E. Jansen, and R. T. Lahey. Computation of incompressible bubble dynamics with a stabilized finite element level set method. Computer Methods in Applied Mechanics and Engineering, 194(42):4565–4587, 2005.

[56] S. Nagrath, K. E. Jansen, R. T. Lahey, and I. Akhatov. Hydrodynamic simulation of air bubble implosion using a FEM based level set approach. Journal of Computational Physics, 215:98–132, 2006.

[57] W. F. Noh and P. Woodward. Slic (simple line interface calculation). In Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics June 28–July 2, 1976 Twente University, Enschede, pages 330–340. Springer, 1976.

[58] S. Osher and R. P. Fedkiw. Level set methods: An overview and some recent results. Journal of Computational Physics, 169:463–502, 2001.

[59] S. Osher and N. Paragios. Geometric level set methods in imaging, vision, and graphics. Springer Science & Business Media, 2003.

[60] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed - algorithms based on hamilton-jacobi formulations. Journal of Computational Physics, 79(1):12–49, 1988.

[61] N. Patankar and D. Joseph. Modeling and numerical simulation of particulate flows by the eulerian–lagrangian approach. International Journal of Multiphase Flow, 27(10):1659–1684, 2001.

[62] J. E. Pilliod and E. G. Puckett. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. Journal of Computational Physics, 199:465–502, 2004.

[63] B. Ray, G. Biswas, A. Sharma, and S. W. J. Welch. CLSVOF method to study consecutive drop impact on liquid pool. International Journal of Numerical Methods for Heat I& Fluid Flow, 23(1):143–158, 2013.

[64] W. J. Rider and D. B. Kothe. Reconstructing volume tracking. Journal of Computational Physics, 141(2):112–152, 1998.

[65] J. M. Rodriguez. Numerical Simulation of Two-Phase Annular Flow. PhD thesis, Rensselaer Polytechnic Institute, July 2009.

[66] J. M. Rodriguez, O. Sahni, R. T. L. Jr., and K. E. Jansen. A parallel adaptive mesh method for the numerical simulation of multiphase flows. Computers & Fluids, 87:115–131, 2013.

[67] E. Roohi, A. P. Zahiri, and M. Passandideh-Fard. Numerical simulation of cavitation around a two-dimensional hydrofoil using vof method and les turbulence model. Applied Mathematical Modelling, 37(9):6469–6488, 2013.

[68] G. Ryskin and L. Leal. Numerical solution of free-boundary problems in fluid mechanics. part 2. buoyancy-driven motion of a gas bubble through a quiescent liquid. Journal of Fluid Mechanics, 148:19–35, 1984.

[69] G. Ryskin and L. Leal. Numerical solution of free-boundary problems in fluid mechanics. part 3. bubble deformation in an axisymmetric straining flow. Journal of Fluid Mechanics, 148:37–43, 1984.

[70] R. Scardovelli and S. Zaleski. Direct numerical simulation of free-surface and interfacial flow. Annual Review of Fluid Mechanics, 31:567–603, 1999.

[71] S. P. Schofield, M. A. Christon, V. Dyadechko, R. V. Garimella, R. B. Lowrie, and B. K. Swartz. Multi-material incompressible flow simulation using the moment-of-fluid method. International journal for numerical methods in fluids, 63(8):931–952, 2010.

[72] J. A. Sethian. Level Set Methods and Fast Marching Methods. Cambridge U. Press, 1999.

[73] J. A. Sethian. Evolution, implementation, and application of level set and fast marching methods for advancing fronts. Journal of Computational Physics, 169:503–555, 2001.

[74] J. A. Sethian and P. Smereka. Level set methods for fluid interfaces. Annual Review of Fluid Mechanics, 35:341–372, 2003.

[75] B. Siebert, C. Maneri, R. Kunz, and D. Edwards. A Four-Field Model and CFD Implementation for Multi-Dimensional, Heated Two-Phase Flows. 2nd International Conference on Multiphase Flows, Kyoto, Japan, 1995.

[76] C. W. Smith, B. Granzow, D. Ibanez, O. Sahni, K. E. Jansen, and M. S. Shephard. In-memory integration of existing software components for parallel adaptive unstructured mesh workflows. In Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale, page 34. ACM, 2016.

[77] M. Sussman. A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. Journal of Computational Physics, 187(1):110–136, 2003.

[78] M. Sussman, A. S. Almgren, J. B. Bell, L. H. Howell, P. Colella, and W. L. Welcome. An adaptive level set approach for incompressible two-phase flows. Journal of Computational Physics, 148:81–124, 1999.

[79] M. Sussman and E. Fatemi. An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. Saim Journal on Scientific Computing, 20(4):1165–1191, 1999.

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

[81] M. Sussman and E. G. Puckett. A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows. Journal of Computational Physics, 162(2):301–337, 2000.

[82] M. Sussman, P. Smereka, and S. J. Osher. A level set approach for computing solutions to incompressible two-phase flows. Journal of Computational Physics, 114:146, 1994.

[83] D. Tatterson, J. Dallman, and T. Hanratty. Drop sizes in annular gas-liquid flows. AIChE Journal, 23(1):68–76, 1977.

[84] C. A. Taylor, T. J. R. Hughes, and C. K. Zarins. Finite element modeling of blood flow in arteries. Comp. Meth. Appl. Mech. Engng., 158:155–196, 1998.

[85] H. Terashima and G. Tryggvason. A front-tracking/ghost-fluid method for fluid interfaces in compressible flows. Journal of Computational Physics, 228(11):4012–4037, 2009.

[86] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. A front-tracking method for the computations of multiphase flow. Journal of Computational Physics, 169(2):708–759, 2001.

[87] A. Tsai, A. Yezzi, W. Wells, C. Tempany, D. Tucker, A. Fan, W. E. Grimson, and A. Willsky. A shape-based approach to the segmentation of medical imagery using level sets. IEEE transactions on medical imaging, 22(2):137–154, 2003.

[88] R. Tsai, S. Osher, et al. Review article: Level set methods and their applications in image science. Communications in Mathematical Sciences, 1(4):1–20, 2003.

[89] S. O. Unverdi and G. Tryggvason. A front tracking method for viscous, incompressible, multifluid flows. Journal of Computational Physics, 100:25, 1992.

[90] Z. Wang, J. Yang, and F. Stern. Comparison of particle level set and CLSVOF methods for interfacial flows. In AIAA 2008-530, AIAA 46th Aerospace Sciences Meeting, 2008.

[91] P. B. Whalley. Boiling, condensation, and gas-liquid flow. Oxford University Press, 1987.

[92] C. H. Whiting. Stabilized finite element methods for fluid dynamics using a hierarchical basis. PhD thesis, Rensselaer Polytechnic Institute, Sep. 1999.

[93] C. H. Whiting and K. E. Jansen. A stabilized finite element method for the incompressible Navier-Stokes equations using a hierarchical basis. International Journal of Numerical Methods in Fluids, 35:93–116, 2001.

[94] X. Yang, A. J. James, J. Lowengrub, X. Zheng, and V. Cristini. An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids. Journal of Computational Physics, 217:364–394, 2006.

[95] K. Yokoi. A practical numerical framework for free surface flows based on CLSVOF method, multi-moment methods and density-scaled CSF model: Numerical simulations of droplet splashing. Journal of Computational Physics, 232:252–271, 2013.

[96] Y. Zhao, H. H. Tan, and B. Zhang. A high-resolution characteristics-based implicit dual time-stepping vof method for free surface flow simulation on unstructured grids. Journal of Computational Physics, 183(1):233–273, 2002.