Development of a Hyperbolic Equation Solver and the Improvement of the OpenFOAM® Two-Phase Incompressible Flow Solver

A thesis submitted in partial fulfilment

of the requirement for the degree of Doctor of Philosophy

Syazana Omar

December 2018

Cardiff University
School of Engineering

To my parents,
for their support.
To my friends,
for being around.
To my cat,
for being a cat.

Abstract

The first part of this thesis proposed new, fully conservative and less oscillatory hyperbolic partial differential equation solvers. Based on the multi-moment method and the Constrained Interpolation Profile Conservative Semi-Lagrangian (CIP-CSL) family of schemes, a new scheme called CIP-CSL3U is introduced to combine with an existing scheme, CIP-CSL3D. Two ENO-like indicators are proposed, which are used to select during runtime a stencil that can efficiently minimise numerical oscillation as well as numerical diffusion. The proposed schemes (CIP-CSL3DU and CIP-CSL3ENO) are validated using various benchmark problems. Discontinuities, as well as smooth solutions, are captured simultaneously with almost no numerical oscillation for non-smooth solutions. Benchmark tests also show that the results are fourth-order accurate for smooth solutions, and can be applied to compressible and incompressible fluid flow problems.

The second part of this work concerns the improvement of the two-phase incompressible flow solver in OpenFOAM. A geometric Level Set method is implemented to couple with a Volume-of-Fluid solver in OpenFOAM. An interface reconstruction algorithm based on cell tetrahedralisation is implemented to work on 2D and 3D unstructured meshes, on serial as well as parallel.

The Coupled Level Set Volume-of-Fluid (CLSVOF) solver is validated against scalar transport problems on various mesh types in 2D and 3D. Results indicate a significant improvement over the standard OpenFOAM solver interFoam and some advant-

age over the newer OpenFOAM solver, interFlow. Mass conservation properties of the VOF method are also retained. The CLSVOF solver is then used to simulate fluid flows with surface tension effects, showing better agreement with experiments and reference solutions compared to standard OpenFOAM solvers. Simulations indicated that CLS-VOF could handle complex fluid flows with surface tension dominance as well as with high density ratios. The calculation of curvature using the Level Set field contributed to the improvement in simulations. A simulation of a liquid jet in a gaseous crossflow also showed reasonable agreement with empirical models with some breakup details captured.

Key words: level-set, volume-of-fluid, ENO, CIP-CSL, OpenFOAM

Acknowledgements

I would like to thank all those who have helped me throughout this four-year study be it professionally or personally.

First and foremost I would like to thank my supervisor Dr Yokoi for his instruction, guidance, and unwavering patience throughout. Thank you to fellow colleagues who made the years bearable; for the valuable discussions, friendship, and camaraderie. Thank you also to my friends and family who stood by in support cheering from the sidelines, and for occasionally feigning interest in my work.

I would also like to thank the research office staff for all their help and assistance, and the team at HPC Wales for their advice and resource.

Finally, I extend my gratitude to my sponsor, Majlis Amanah Rakyat who provided financial support throughout this study without which this work would not have been possible.

Contents

List of Publications

The work introduced in this thesis is based on the following publications.

- Qijie Li, Syazana Omar, Xi Deng, and Kensuke Yokoi (2017). *Constrained Interpolation Profile Conservative Semi-Lagrangian Scheme Based on Third-Order Polynomial Functions and Essentially Non-Oscillatory (CIP-CSL3ENO) Scheme.* Communications in Computational Physics. Vol. 22 (3) pp. 765 - 788.
- Syazana Omar (2016). *A 3rd order ENO-like multi-moment method for solving hyperbolic conservation laws.* Proceedings of 24th Conference on Computational Mechanics (ACME-UK), 31 March - 1 April 2016.

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Most problems in Computational Fluid Dynamics are governed by hyperbolic partial differential equations (HPDE), so the solution of HPDEs lies as the basis of numerical algorithms for solving fluid flow problems. However, to calculate the solution numerically is not always straightforward; around discontinuities, one will obtain poor numerical results using methods that otherwise work well in smooth regions [18]. The numerical oscillations that occur around discontinuities (Fig. 1.1(b)) pollute the solution by producing non-physical extrema which might eventually cause the simulation to collapse. One can suppress numerical oscillations by resorting to first-order methods which are strictly monotone but at the cost of smearing out the solution as in Fig. 1.1 (a). The development of numerical solvers that can handle discontinuities without giving way to numerical diffusion is, therefore, a field of active interest for many researchers in numerical methods and forms the first part of this thesis.

The second part of this thesis concerns the solving of two-phase incompressible fluids flows. A two-phase flow is a system in which two different phases of fluids coexist. These fluids can be gas and liquid, or liquid and liquid, where the former is more common. These flows are commonly observed in nature as well as in practical applications. Some examples of naturally-occuring phenomena are falling raindrops, waterfalls, dew on leaves, and wind-driven waves. They are also found in a wide range of industrial applications; for example, the study of two-phase flows are central for understanding phenomena such as droplet streams of inkjet printers [19], the aerosolisation of certain medications [20], the steam-water interaction in nuclear reactors [21], and the design

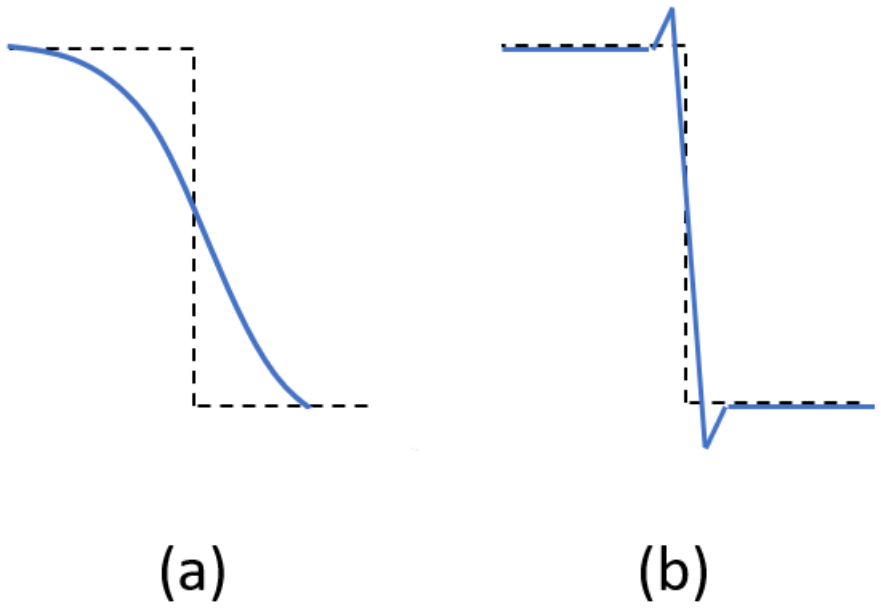


Figure 1.1: (a) Numerical diffusion over a square wave, and (b) numerical oscillation near a discontinuity.

of fuel injectors [22].

Fuel injector design is an especially notorious problem; in 2016, Rolls Royce invested £1.3 billion on research and development [23], including on the study of gas turbine systems. Gas turbine fuel injection occurs in a highly turbulent swirling environment, where large-scale mixing is induced by poorly understood aerodynamic phenomena and two-phase fluid mechanics [24]. Furthermore, in order to reduce NOx pollution, Rolls Royce has also invested in the RR CLEEN II Low NOx Combustor program to improve combustor performance [25]. This also involves advanced fuel injection capabilities and by extension, deep understanding of spray characteristics. However, due to the challenging nature of the environments in which engine conditions operate, experimental observation proves difficult. For diesel sprays, for example, the experimental characterisation of the initial stage of jet formation and primary breakup under realistic engine conditions occur in harsh settings and the process is highly transient

in nature, with elevated velocities and microscopic scales. Ricardo, in collaboration with Brighton University, employs advanced diagnostic techniques for use in model validation, all towards furthering the understanding of air flows, sprays, and combustion processes [26]. Understanding the effects of spray characteristics are critical for the accurate prediction of combustion, and this is also vital to ensure future emissions regulations are met.

With these examples in mind, it is clear that a robust and accurate solver to simulate two-phase flows would always prove useful for industrial applications as well as research. Modelling two-phase flows, however, can be complex. The presence of interfacial surfaces introduces challenges in the physical and numerical formulation of the problem [27]. Topological changes which commonly occur in such flows can be severe (such as the case of a spray formed by a fuel injector in Fig. 1.2), requiring sophisticated methods of interface tracking or interface capturing. Therefore, the development of solvers that can handle complex interface deformation is one of the main areas of interest where two-phase flow modelling is concerned.

1.1 Research aim and objectives

To address the issues outlined in the previous section, this work firstly aims to propose a more accurate fluid advection solver that improves the way discontinuities in fluid flow properties are handled during the discretisation process. Secondly, this work aims to develop and implement a more accurate interface capturing scheme in order to make improvements in the quality of two-phase flow simulations. It is intended that the research findings contribute to the overall improvement of simulation tools to capture complex interface deformation that occurs in many fluid phenomena found both in nature and industry.

The above aims raise the following core objectives:

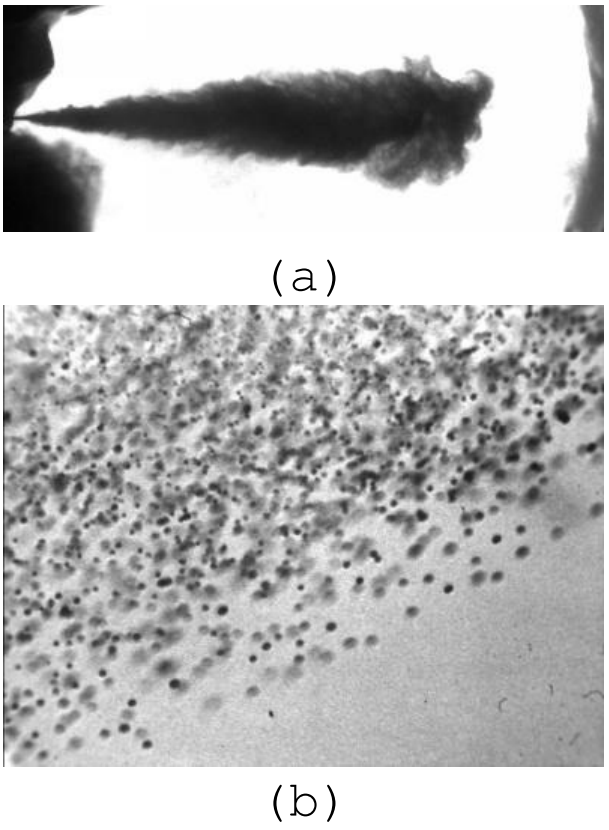


Figure 1.2: (a) Spray formed by a diesel fuel injector and (b) the droplets formed at the edge of the fuel spray. Pictures reproduced from Helsinki University of Technology [1].

1. to develop a hyperbolic partial differential equation solver that is able to capture discontinuities and smooth solutions simultaneously well with minimal numerical oscillation and diffusion,

2. to improve the two-phase incompressible solver within the open-source CFD code repository OpenFOAM® by implementing an explicit interface capturing scheme,

3. to implement an existing dynamic contact angle model into the improved two-phase solver,

4. and to validate the two-phase incompressible flow solver against various bench-

mark tests and complex fluid flow problems.

1.2 Thesis outline

This thesis is organised into two main parts. The first part concerns the development of a multi-moment method to solve hyperbolic partial differential equations. The second part describes the implementation and validation of an explicit interface capturing scheme for two-phase flows and its possible applications in complex fluid flows. The breakdown of the topics addressed in each chapter is as follows:

The literature review goes over the governing equations used in this work, and describes the fluid solvers and numerical models on which the work in this thesis are based. A general overview is given to identify issues that are addressed in this work.

The subsequent chapter proposes a novel hyperbolic equation solver based on a multi-moment method for better handling of sharp discontinuities in fluid properties. The proposed method aims to minimise numerical oscillations near discontinuities whilst maintaining a sharp profile.

The next chapter describes the proposed implementation of a fully 3D geometric Coupled Level Set Volume of Fluid method on unstructured meshes using OpenFOAM. The scheme is validated using scalar transport problems in 2D and 3D on structured and unstructured meshes and compared with some existing solvers.

The following chapter then describes the implementation of a dynamic contact angle for the proposed CLSVOF scheme. The implementation is validated against benchmark tests and some challenging applications in 2D and 3D, including the simulation of binary droplet collisions and the simulation of a liquid jet in a gaseous crossflow.

The final chapter summarises the achievements presented in this work with some suggestions for future improvements.

Chapter 2

Literature Review

In this chapter, the development of two-phase incompressible fluid solvers are considered with emphasis on the following interface capturing schemes; the Volume of Fluid method, the Level Set method, and the Coupled Level Set Volume of Fluid method. The discretisation strategy employed in the open source CFD toolbox OpenFOAM is discussed. This is followed by an introduction to the Constrained Interpolation Profile Conservative Semi-Lagrangian family of hyperbolic equation solvers.

2.1 Governing equations

The governing equations of an incompressible, immiscible, isothermal flow can be written in the form of conservation of mass;

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{2.1}$$

and of the conservation of momentum;

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla P + \nabla \cdot \tau_s + \rho \mathbf{g} + F_\sigma, \tag{2.2}$$

where ρ is density, $\mathbf{u}$ is velocity, P is the pressure, F_σ is the volumetric surface tension force, and $\mathbf{g}$ is the gravitational acceleration. τ_s is the viscous stress tensor defined as $\tau_s = 2\mu(0.5[(\nabla \mathbf{u}) + (\nabla \mathbf{u})^T])$ where μ is the viscosity.

For incompressible fluids, the velocity divergence is zero;

$$\nabla \cdot \mathbf{u} = 0. \tag{2.3}$$

2.2 Interface capturing schemes

The simulation of two-phase flows requires a technique to identify the boundary between the two fluids, as this boundary is not known beforehand [28]. Among the phenomena that need to be handled are topological changes of the interface, discontinuities, coalescence, and breakup. Several interface modelling techniques have been developed to tackle complex fluid flows, and the numerical methods to accurately track or capture the interface between two fluids can generally be split into two main categories, which are Lagrangian and Eulerian.

In Lagrangian methods, the grid follows the fluid, whose interface is represented using marker-points. The Navier-Stokes equations are then solved on the grid. For example, Brackbill, Kothe, and Ruppel proposed a particle-in-cell (PIC) method using fully Lagrangian particles to eliminate convective transport [29]. Despite the promising accuracy, the non-automatic handling of topological change renders it very complex to implement in 3D. Despite the difficulty, Johnson and Tezduyar successfully [30] developed a tool for 3D simulations of fluid-particle interactions with fairly impressive results. There are also pure Lagrangian schemes where no mesh is used and the flowfield is evaluated at the Lagrangian points [31] [32]. However, true continuity enforcement may be difficult [33].

Eulerian approaches are more commonly taken, where they can be further subdivided into non-fixed and fixed grid methods. Some of the non-fixed grid Eulerian schemes that have been developed include the boundary-fitted grid proposed by Ryskin and Leal [34] where the grids are free to move with the interface motion and the Lattice Boltzmann method which minimises the free energy functional to naturally capture the interface [35] [36]. However, these methods are best suited for relatively simple geometries only.

This brings us to the most popular approach yet; Eulerian schemes with fixed grid. Among these is the marker-and-cell (MAC) method where marker particles are used to identify the fluids [37]. The Volume-of-Fluid method [38] uses a volume fraction function to indicate the quantity of each fluid in each cell and is very popular among researchers [39]. However, it may be difficult to maintain a sharp interface in MAC and VOF schemes. The Level-Set method [40] which uses a signed distance function address this issue by naturally representing the interface using the 0-contour field, but at the cost of mass conservation [41].

To address the mass conservation issues of the Level Set method, some hybrid schemes have been proposed. For example, Enright et al. proposed a method which is a hybrid of the Level Set and Lagrangian particle schemes [42]. A more common hybridisation is between the Level Set and Volume of Fluid methods [43], [44], [45], [46], which is the main focus of this work. In the following sections, the methodologies of the LS, VOF, and CLSVOF methods are reviewed.

2.2.1 Level set method

The level set formulation was first proposed by Osher and Sethian in 1988 [40] as a relatively simple and versatile method for analysing the motion of an interface in two or three dimensions. In 1994, Sussman, Smereka, and Osher [47] proposed a Level Set approach for computing solutions of incompressible two-phase flows.

Typically the Level Set function is represented as a smooth field and denoted using the signed distance ϕ, where

$$|\nabla\phi| = 1 \tag{2.4}$$

is satisfied. As the interface is represented implicitly by the iso-contour $\phi = 0$, the sign of the function represents different fluid phases and the scalar value represents the normal distance from the interface. The usual convention is to set the negative value in the less dense liquid, and the positive in the denser one (Fig. 2.1).

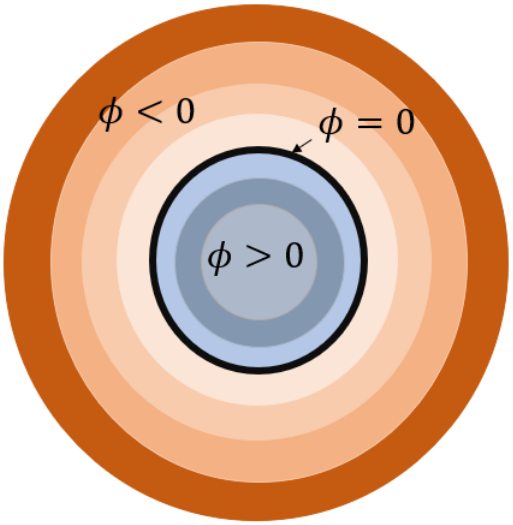


Figure 2.1: Sketch of a Level Set field with $\phi = 0$ at the fluid interface

A smoothed Heaviside function is generated in terms of ϕ;

$$H(\phi) = \begin{cases} 0 & \text{if } \phi < \epsilon \\ \frac{1}{2}[1 + \frac{\phi}{\epsilon} + \frac{1}{\pi}\sin(\frac{\pi\phi}{\epsilon})] & \text{if } |\phi| \leq \epsilon \\ 1 & \text{if } \phi > \epsilon, \end{cases} \tag{2.5}$$

where ϵ is the thickness of the transition region between the liquid and gas phases. The Heaviside function is used to define the physical properties density ρ and viscosity μ of the fluid. The values ρ and μ are found as;

$$\rho = \rho_L H(\phi) + \rho_G(1 - H(\phi)), \tag{2.6}$$

$$\mu = \mu_L H(\phi) + \mu_G(1 - H(\phi)), \tag{2.7}$$

where ρ_L and ρ_G are the densities of liquid and gas, and μ_L and μ_G are the viscosities of liquid and gas. The main advantage of the level-set method is that the topological changes of the evolving front are handled naturally, as it is simply the zero-contour of

the level set field. This front is able to break and merge with the evolution of time. Geometric quantities such as the normal vector $\mathbf{n}$ and curvature $\mathcal{K}$ can be easily approximated as

$$\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|} \text{ and } \mathcal{K} = -\nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \tag{2.8}$$

which is used to calculate the volumetric surface tension force $\mathbf{F}_\sigma$;

$$\mathbf{F}_\sigma = \sigma\mathcal{K}(\phi)\delta(\phi)\nabla\phi. \tag{2.9}$$

δ is the Dirac function used to limit $\mathbf{F}_\sigma$ to a narrow band around the interface, defined as

$$\delta(\phi) = \begin{cases} 0 & \text{if } |\phi| > \epsilon \\ \frac{1}{2}(1 + \cos(\frac{\pi\phi}{\epsilon})) & \text{if } |\phi| \leq \epsilon. \end{cases} \tag{2.10}$$

The following equation;

$$\phi_t + (\mathbf{u} \cdot \nabla)\phi = 0, \tag{2.11}$$

propagates the zero level-set of ϕ in time. However, solving Eq. (2.11) using low-order convection schemes often leads to a smeared solution. It is also noted in [45] that ϕ would not remain a true distance function after Eq. (2.11) is solved (i.e. $|\nabla\phi| \neq 1$), so there is a necessity to reinitialise ϕ so that it continues to be a distance function. This is done as follows;

$$\frac{\partial\phi}{\partial\tau} = S(\phi_0)(1 - |\nabla\phi|) \tag{2.12}$$

where τ is an artificial time, and $\mathrm{S}(\phi_0)$ is the sign of the initial level set function S_0, usually taken to be

$$S_0 = \frac{\phi}{\sqrt{\phi_0^2 + 10^{-5}}} \tag{2.13}$$

for the purposes of stability, where ϕ_0 is the LS value at the current time step before reinitialisation. Here we define the $|\nabla\phi|$ in Eq. (2.12) as

$$|\nabla\phi| = \frac{|\nabla\phi|^2}{|\nabla\phi|} = \frac{\nabla\phi}{|\nabla\phi|} \cdot \nabla\phi. \tag{2.14}$$

Eq. (2.12) is solved until it reaches steady state, which does not require many steps if the original level set field is already close to the distance function.

Despite the ability of the LS method to capture the fluid interface smoothly (and hence its robust calculation of interface normal vectors), it is known to be not conservative; the total mass or volume confined by the interface may not be preserved.

2.2.2 Volume of fluid method

The Volume of Fluid (VOF) method was first introduced by Noh and Woodward in 1976 [48], and later employed by Hirt and Nichols [38]. A highly popular scheme, it is available in commercial and open-source softwares such as OpenFOAM [49], ANSYS Fluent [50], Gerris [51], and FLOW-3D [52]. A VOF function α is defined such that it is unity at any point occupied by fluid, and zero otherwise. The average value of α in a cell would then represent the fractional volume of that cell occupied by fluid, as demonstrated in Fig. 2.2.

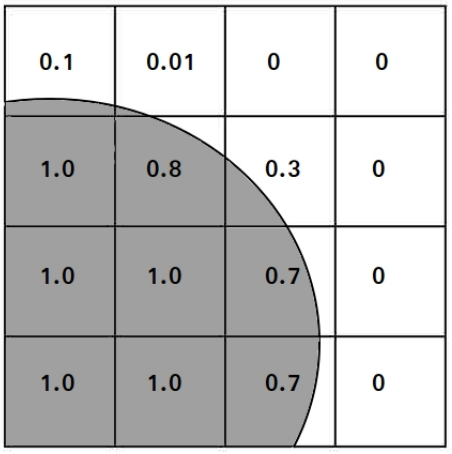


Figure 2.2: Schematic of a fluid distribution in a 2-D Cartesian grid with its accompanying indicator α values.

Cells with α values between 0 and 1 must therefore contain an interface. The following equation is solved;

$$\alpha_t + \nabla \cdot (\mathbf{u}\alpha) = 0, \tag{2.15}$$

and the physical properties of the immiscible fluids are calculated as such;

$$\rho = \rho_L \alpha + \rho_G (1 - \alpha), \tag{2.16}$$

$$\mu = \mu_L \alpha + \mu_G (1 - \alpha). \tag{2.17}$$

VOF methods have benefited from continuous improvement over decades due to its popularity among researchers in the field of interface capturing. Some of the reasons for their widespread usage include, as described in [39];

1. the mass is conserved naturally due to the development of an advection algorithm based on a discrete representation of the conservation law (Eq.2.15),
2. extension from 2D to 3D is relatively straightforward,
3. domain decomposition for parallel implementation is relatively simple as the α values in a cell only depends on the α values in its neighbouring cells.

Since only the volume fraction α is known, approximating the exact interface in each cell may be difficult. The available VOF methods are generally divided into two categories; algebraic VOF methods and geometric VOF methods.

In algebraic VOF methods, explicit reconstruction of the interface in each cell is not necessary. Some examples of this type of scheme is the one proposed by Nichols and Hirt [53], Lafaurie et al. [54], and Ubbink and Issa [55]. Using these methods, the interface is usually represented by a 'sharpening', using modified convection schemes or additional numerical terms. However, the interface is usually still smeared over several cells.

The geometric VOF schemes, meanwhile, usually take an extra step to explicitly reconstruct the interface. Some of the earliest and simplest types of geometric VOF is the Simple Line Interface Calculation (SLIC) (Fig. 2.3 (c), (d)) by Noh and Woodward [48] and the SOLA-VOF by Hirt and Nichols [38]. These algorithms construct

the interface in a multiphase cell by a segment aligned with the grid. However, these reconstruction methods are at best first-order accurate and tend to generate a lot of flotsam even for cases with simple velocity fields.

A more accurate geometric VOF interface recontruction technique is the Piecewise Linear Interface Construction (PLIC) method (Fig. 2.3 (b)) such as those proposed by Ashgriz and Poo [56], Youngs [57], Gueyffier et al. [58], and Pilliod and Puckett [59]. Typically, the surface is represented by a sequence of polygons in a 3D cell, with some discontinuity in between. PLIC methods can be second-order accurate but are highly cumbersome to implement on unstructured 3D meshes.

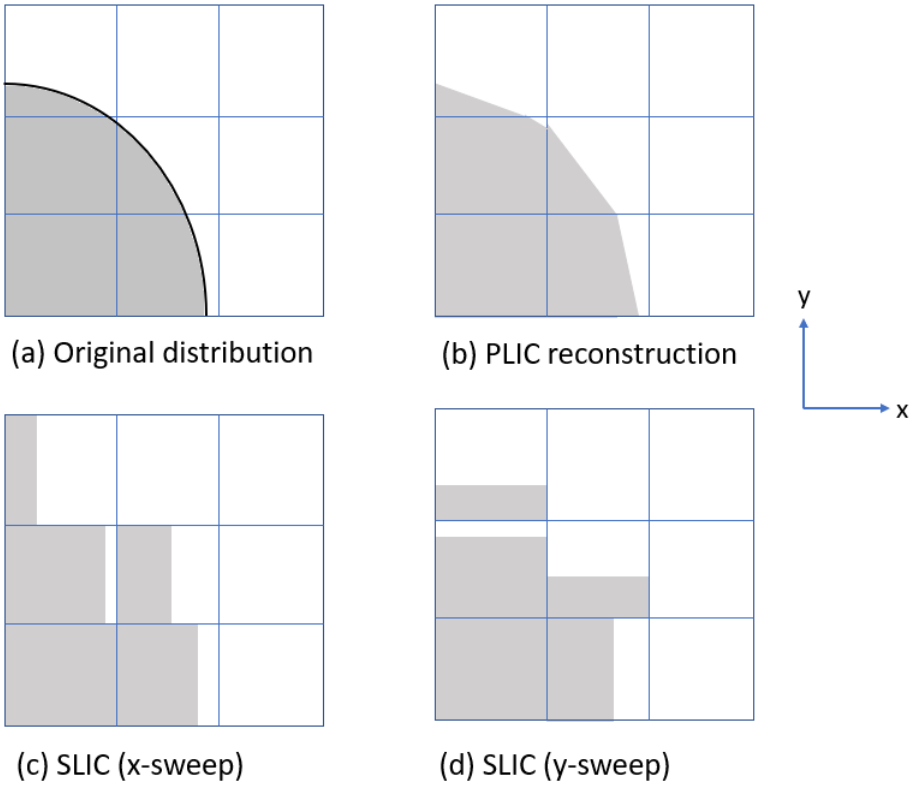


Figure 2.3: Comparison of VOF different techniques for predicting the fluid distribution.

In VOF methods, the volumetric surface tension force in the momentum equation is commonly calculated using the Continuum Surface Force (CSF) model as proposed by

Brackbill et al. [60], as

$$\mathbf{F}_\sigma = \sigma \mathcal{K}(\alpha) \nabla \alpha \tag{2.18}$$

where $\mathcal{K}(\alpha)$ is the interface curvature calculated using α. The curvature is represented as;

$$\mathcal{K} = -\nabla \cdot \hat{\mathbf{n}}_c. \tag{2.19}$$

where $\hat{\mathbf{n}}_c$ indicates the unit interface normal, also calculated using the α field;

$$\hat{\mathbf{n}}_c = \frac{\nabla \alpha}{|\nabla \alpha|}. \tag{2.20}$$

In this work, two methods are considered for the solution of Eq. (2.15); the interface is propagated using the algebraic *interFoam* solver (a compressive VOF method) and a geometric technique implemented proposed by Roenby et al. [2], *isoAdvector*. These methods are explained as follows.

2.2.2.1 interFoam

The open-source CFD package OpenFOAM [49] features an extensive range of features, enabling it to solve various complex fluid flows such as flows with chemical reactions, turbulence, heat transfer, and multiphase flows. Compared to the popular commercial solver FLUENT, OpenFOAM has a significantly steeper learning curve due to its lack of a graphical user interface (GUI) and its almost infinite options for manipulation. At the same time, this flexibility makes it a very powerful tool for expert users, as it can be freely modified to suit the user's needs.

interFoam is a VOF-based two-phase incompressible, immiscible fluid solver available in OpenFOAM. In this implementation, the advection equation is reformulated by Weller by adding a compressive term to retain conservation, convergence, and boundedness [61];

$$\alpha_t + \nabla \cdot (\mathbf{u}\alpha) + \nabla \cdot (\mathbf{u}_c \alpha \beta) = 0, \tag{2.21}$$

where $\beta = 1 - \alpha$ and $\mathbf{u}_c = \mathbf{u}_L - \mathbf{u}_G$, which is the relative velocity between the liquid and gas or the compressive velocity [62], and L and G stand for Liquid and Gas.

This compressive velocity is only considered in the region of the interface. To avoid dispersion, it is defined only around the interface which is achieved by multiplying it with $\nabla\alpha/|\nabla\alpha|$. A compression factor c_α can also be used to increase compression;

$$\mathbf{u}_c = \min\left(c_\alpha|\mathbf{u}|, \max\left(|\mathbf{u}|\right)\right)\frac{\nabla\alpha}{|\nabla\alpha|}, \tag{2.22}$$

where max $|\mathbf{u}|$ is the maximum speed anywhere inside the domain. $c_\alpha = 0$ indicates no compression, with $c_\alpha = 2$ as the maximum compression. In this work, the compression factor is set as $c_\alpha = 1$ as recommended by Deshpande [63], as increasing it or decreasing it exacerbates errors in interface curvature and smearing.

interFoam is a widely used two-phase incompressible flow solver in the research community and has been investigated by various parties. Deshpande et al. [63] performed a comprehensive evaluation of the interFoam solver and found that while generally robust, the curvatures computed by the solver may converge to a value different from the analytical value. The disruptive effects of spurious currents produced by interFoam was noted in this regard. Since the interface is only implicitly captured, the interface location, normal, and curvature are also only known implicitly. Another noted issue with interFoam is the difficulty to maintain a sharp and accurate interface compared to methods that use explicit interface capturing, which prompts the discussion of $interFlow$ in the next section.

2.2.2.2 interFlow

In 2016, Roenby, Bredmose, and Jasak [2] proposed a new method for interface capturing in OpenFOAM called the *isoAdvector* method, motivated by coastal and marine simulations involving violent breaking waves. In this scheme, an explicit 'isosurface' is reconstructed for each interface cell for each time step (Fig. 2.4(a)), which ensures that it does not suffer from the same diffusive interface representation as interFoam. Using this isosurface, the motion of the face-interface intersection line (the line created as the fluid interface plane in a cell intersects the cell face) is modelled within a

time step to obtain an accurate estimate for the volume of fluid transported across each face. Fig. 2.4(b) demonstrates how an isosurface moves through a cell within a single time-step. This will now be explained further.

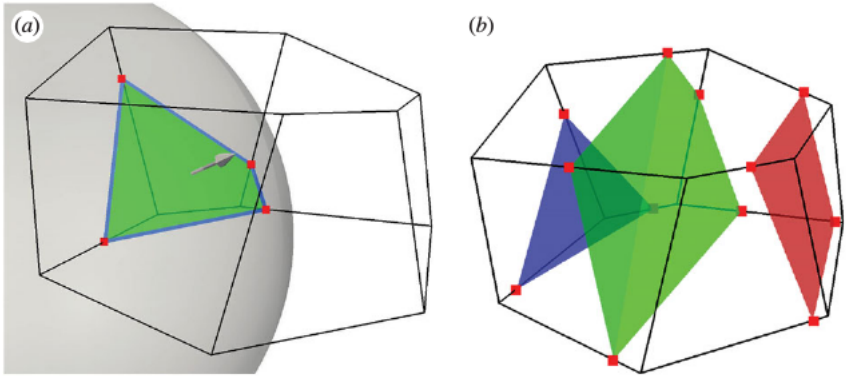


Figure 2.4: (a) A surface cutting through a cell, with dots signifying cutting points on cell face. The surface in the cell is the isoface. (b) The isoface being propagated at three different intermediate times τ within a time step. Figure reproduced from [2].

Consider a domain where resides a surface S. Since this work concerns two incompressible and immiscible fluids (fluid L and fluid G), surface S denotes the separation between the two fluids. These fluids are advected in a continuous velocity field $\mathbf{u}(\mathbf{x}, t)$ defined throughout the domain. The full two-phase incompressible solver using isoAdvector for the interface advection is known as the **interFlow** method. Since isoAdvector is focused on the advection of the interface, assume for now that $\mathbf{u}(\mathbf{x}, t)$ is known throughout the domain for all time t. The evolution of the interface $S(t)$ is first represented in terms of density $\rho(\mathbf{x}, t)$;

$$\frac{d}{dt}\int_V \rho(x,t)\, dV = -\int_{\partial V} \rho(x,t)\, \mathbf{u}(x,t)\cdot d\mathbf{S} \tag{2.23}$$

where V is an arbitrary volume, ∂V is its boundary, and $d\mathbf{S}$ is the differential area vector that points outwards of V. This equation represents, in words;

Instantaneous rate of change of total mass in V equals the instantaneous flux of mass through boundary S.

The solution does not depend on the values of ρ_L and ρ_G in a pure advection problem with a set velocity field, so a Heaviside function $H(\mathbf{x}, t)$ is used as a simplified indicator, where

$$H(x,t) \equiv \frac{\rho(x,t) - \rho_G}{\rho_L - \rho_G}. \tag{2.24}$$

For cells fully occupied by fluid L, H is unity, and for cells fully occupied by fluid G, H is zero. The computational domain is divided into cells hereafter denoted as C_i, where $i = 1, 2, ..., N_V$. Two cells next to each other will have a shared boundary (or internal face). Faces are labelled $j = 1, ..., N$ and the surface of face j is $\mathcal{F}_j$. The boundary of cell i can now be represented as a list B_i, which contains all the labels of the faces that belong to its boundary δV_i.

Substitute Eq. 2.24 into Eq. 2.23, where we integrate over the volume of cell i;

$$\frac{d}{dt}\int_{C_i} H(x,t)dV = -\sum_{j\in B_i} s_{i,j}\int_{F_i} H(x,t)\mathrm{u}(x,t)\cdot d\mathbf{S}. \tag{2.25}$$

Face j has its own orientation to identify the direction of $d\mathbf{S}$, so $s_{ij} = +1$ or -1 such that s_{ij} points out of cell i for face j. The volume fraction of fluid L in cell i is defined as

$$\alpha_i(t) \equiv \frac{1}{V_i}\int_{C_i} H(x,t)dV, \tag{2.26}$$

where V_i is the volume of cell i. Inserting Eq. 2.26 into Eq. 2.25 gives

$$\alpha_i(t+\Delta t) = \alpha_i(t) - \frac{1}{V_i}\sum_{j\in B_i} s_{ij}\int_t^{t+\Delta t}\int_{F_i} H(x,\tau)\,\mathbf{u}(x,\tau)\cdot d\mathbf{S}\,d\tau \tag{2.27}$$

The time integral on the RHS is the total volume of fluid L transported across face j in the interval $[t, t+\Delta t]$, so it is a fundamental quantity to be estimated to advance in time the quantity α_i (and as a result, the interface S). This is now called $\Delta V_j(t, t+\Delta t)$, where;

$$\Delta V_j(t, t+\Delta t) \equiv \int_t^{t+\Delta t}\int_{F_i} H(x,\tau)\mathbf{u}(x,\tau)\cdot d\mathbf{S}\,d\tau. \tag{2.28}$$

Substituting 2.28 into 2.27 gives

$$\alpha_i(t+\Delta t) = \alpha_i(t) - \frac{1}{V_i}\sum_{j\in B_i} s_{ij}\Delta V_j(t, t+\Delta t). \tag{2.29}$$

In finite volume methods, the velocity is represented as cell-averaged values (i.e. at the cell centre);

$$\mathbf{u}_i(t) = \frac{1}{V_i} \int_{C_i} \mathbf{u}(\mathbf{x}, t)\, dV. \tag{2.30}$$

Another representation of the velocity field in OpenFOAM is the volumetric fluxes across cell faces;

$$\mathbf{F}_j(t) = \int_{F_i} \mathbf{u}(x, t) \cdot d\mathbf{S}. \tag{2.31}$$

Knowing the values α_i, $\mathbf{u}_i$, and $\mathbf{F}_j$, isoAdvection aims to estimate the fluid L volume transport ($\Delta V_j(t, t+\Delta t)$) across a face in the time interval $[t, t+\Delta t]$. Two assumptions are made in this work;

- the local radius of curvature is larger than the cell size i.e. the interface is well-resolved,
- the velocity field is constant in time between $[t, t + \Delta t]$, i.e. $\mathbf{u}(\mathbf{x}, \tau) \approx \mathbf{u}(\mathbf{x}, t)$.

In Eq. 2.27, the $\mathbf{u}$ on face $\mathcal{F}_j$ dotted with differential face normal vector $d\mathbf{S}$ can be approximated it terms of volumetric face flux $\mathbf{F}_j$, as

$$\mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S} \approx \frac{\mathrm{F}_j(t)}{|\mathbf{S}_j|} dS \text{ for } \mathbf{x} \in \mathcal{F}_j \tag{2.32}$$

where $dS \equiv d|\mathbf{S}|$ and cell face normal is given as

$$\mathbf{S}_j \equiv \int_{\mathcal{F}_j} d\mathbf{S} \tag{2.33}$$

With this in mind, Eq. 2.27 is solved by substituting in Eq. 2.33;

$$\Delta V_j(t, t + \Delta t) \approx \frac{\mathbf{F}_j(t)}{|\mathbf{S}_j|} \int_t^{t+\Delta t} \int_{\mathcal{F}_i} H(x, \tau)\, dS\, d\tau \tag{2.34}$$

where $\int_{\mathcal{F}_i} H(x, \tau)\, dS$ is then the *instantaneous area of face j submerged in fluid L* i.e. underneath the interface plane. This is referred to as $A_j(\tau)$. Now Eq. 2.34 is written as

$$\Delta V_j(t, t + \Delta t) \approx \frac{\mathbf{F}_j(t)}{|\mathbf{S}_j|} \int_t^{t+\Delta t} A_j(\tau)\, d\tau. \tag{2.35}$$

When velocity is constant in space and time, Eq. 2.35 becomes exact. If the mesh is sufficiently fine compared to the velocity field gradients, and the time steps are small enough compared to the temporal variations of velocity, the error from this approximation becomes immaterial.

The time integral of $A_j(\tau)$ in Eq. 2.35 is calculated analytically to obtain the estimate of total volume of fluid L transported across face j in the interval $[t, t + \Delta t]$.

Investigations by Roenby et al. show that the interFlow method has considerable advantage over interFoam for interface sharpness. However, since the conception of this scheme is for the application of large ocean waves where surface tension effects are negligible, the mean curvature estimation is not greatly improved compared to interFoam.

2.2.2.3 Solution procedure

The procedure of the OpenFOAM VOF solver for two-phase incompressible flows is summarised as follows;

1. Initialise the variable fields (α, P, $\mathbf{u}$, ρ).
2. Solve the volume fraction advection equation Eq. (2.15).
3. Calculate the interface normal (Eq. 2.20) and curvature (Eq. 2.19).
4. Update the physical properties of the mixture, density (Eq. 2.16) and viscosity (Eq. 2.17).
5. Perform the velocity-pressure correction loop (PISO) (details in Appendix 6.2).
6. Go to the next time step from Step 2.

2.2.3 CLSVOF

Combining the previous two discussed schemes (VOF and LS) has been a popular field of research in the past decade. This arises from the complementary nature of each of the schemes (Fig. 2.5); by combining them, one gains the mass conservation properties of VOF as well as the interface sharpness of the LS method. One of the earliest proposed Coupled Level-Set Volume of Fluid method (CLSVOF) is by Sussman and Puckett [45], followed by various other researchers [46] [43] [44] [64] [65].

	Volume of Fluid	Level Set
Field representation	Phase fraction α: Phase 1: α = 1 Phase 2: α = 0	Signed distance value Φ: Phase 1: Φ > 0 Phase 2: Φ < 0
Interface representation	0 < α < 1	Φ = 0
Transport	$\frac{\partial \alpha}{\partial t} + \nabla \cdot (U\alpha) = 0$	$\frac{\partial \Phi}{\partial t} + \nabla \cdot (U\Phi) = 0$
Mass conservation	Yes	No
Interface quality	Diffusive	Sharp

Figure 2.5: Summary of VOF and LS properties

In this section we summarise the CLSVOF scheme as implemented by Sussman in [45] for structured meshes in 2D. The fields are first initialised with the LS (ϕ^0) as a signed distance function from the fluid interface which is represented by its zero-contour. The VOF field (α^0) is initialised from ϕ^0 using the Heaviside function as in Eq. (2.5).

The LS and VOF fields are transported as follows;

$$\phi_t + \nabla \cdot (\mathbf{u}\phi) = 0, \tag{2.36}$$

and

$$\alpha_t + \nabla \cdot (\mathbf{u}\alpha) = 0. \tag{2.37}$$

The LS fluxes are calculated using its values at cell centres and velocity values at cell faces. The fluxes of the VOF function is calculated in terms of the reconstructed piecewise linear equation of the LS function;

$$\phi^R_{i,j}(r,z) = A_{i,j}(r - r_i) + B_{i,j}(z - z_j) + C_{i,j}, \tag{2.38}$$

which is unique for each cell. x, y represent the Cartesian coordinates, (i, j) indicate the cell ID index. The coefficients A and B are constants that define the gradient of the linear reconstruction equation, and C determines the position of the interface line from the cell centre. The coefficients A, B, and C are found such that $\phi^R_{i,j}(r, z)$ represents the best fit line for the zero level set in that cell, which means C must be adjusted such that the interface line cuts the cell to give the same volume as given by the VOF value α in that cell.

After the new values $\phi*$ and α^1 are obtained, the LS field must undergo a reinitialision process to maintain its property as a signed distance function. Outside a certain thickness region that defines a band around the interface, the volume fraction is truncated. For a band of cells around the interface, their respective LS values are calculated geometrically such that their exact distance to the zero-contour interface is found. For all other cells outside this band, a set value is given for their LS value, with their sign maintained as this reflects the fluid phase that occupies that cell. Thus the value ϕ^1 is found for the new time step.

There have been many variations of the CLSVOF implementation since its inception. Son and Hur [44] proposed a geometric reconstruction of the interface using an additional geometrical parameter which is the furthest distance between cell corners that are occupied by the liquid and interface. The interface position is then calculated using this parameter. The liquid volume fraction in the interface cell is satisfied by reconstructing the interface using the aforementioned parameter and the interface normal.

Menard et al. [43] implemented the CLSVOF method without using geometrical reconstruction, which is considered the major technical challenge in the implementation

of CLSVOF. They proposed an analytical procedure to calculate the constants in Eq. (2.38). The reinitialisation is then performed iteratively, as proposed by Sussman in an earlier work [66].

Albadawi et al. [67] proposed S-CLSVOF (Simplified CLSVOF) as an improvement of the OpenFOAM solver, interFoam. This is an extension of the Kunkelmann implementation [68]. In this proposal, only the VOF field is advected and the LS field is set to the 0.5-contour of the VOF field. The curvature calculation is then calculated using the LS field, which yields some improvements in surface tension dominant cases. However, this implementation is more focused on structured meshes which may limit its applications to simpler geometries.

Dianat et al. [69] implemented the CLSVOF scheme into OpenFOAM as an extension of the interFoam solver. The interface is found iteratively using a 'clip and cap' approach by Ahn and Shashkov [70] and the calculation of face flux in interFoam is improved; instead of interpolating the α value to the cell face, the exact area of the face that is occupied by the fluid is used instead to calculate flux. This implementation shows good performance on non-orthogonal meshes.

Certainly the coupling of VOF with LS is not a novel idea; in the last few years, CLSVOF methods have started to appear as a good alternative to either LS or VOF alone as the drawbacks of each scheme individually can be addressed by coupling them. While CLSVOF does incur a computational cost compared to using LS or VOF individually, the increase in computational power in recent years as well as the availability of high performance computing systems have softened this impact, making accurate but computationally-intensive schemes such as the geometrical CLSVOF method more attractive.

2.3 Finite volume discretisation

To solve partial differential equations in fluid models computationally, a discretisation process is required. Common discretisation methods are the Finite Difference method (FD) [71] [72], the Finite Volume method (FV) [73] [74], and the Finite Elements method (FE) [75]. For fluid dynamic problems, the Finite Volume method is a natural choice owing to its conservative properties and was introduced in the early 1970s by McDonald [76], and MacCormack and Paullay [77]. In this method, the governing equations are discretised by dividing the continuum into a number of arbitrary, polygonal control volumes. The integral formulation of conservation laws are discretised directly in space, the advantage of which is that it enforces the conservation of quantities. There are two approaches of defining the shape and position of the control volume with respect to the grid cells [78]:

- Cell-centered scheme, where the flow quantities are stored at the cell-centroid,
- Cell-vertex scheme, where the flow quantities are stored at grid points. The control volume would either be all cells sharing this point (overlapping control volumes), or some volume centered around the point (dual control volumes).

Our framework is based on the OpenFOAM® open-source code, which is a cell-centered FV formulation. In this section, the discretisation procedure as applied in OpenFOAM is described. This includes the details of the solution domain, and the spatial and temporal discretisation schemes used.

2.3.1 The solution domain

The computation domain denoted V_M is divided into m control volumes (CV) of any convex polygonal shape on the condition that these do not overlap each other. This creates an arbitrary unstructured mesh that completely fills the entire domain, and all

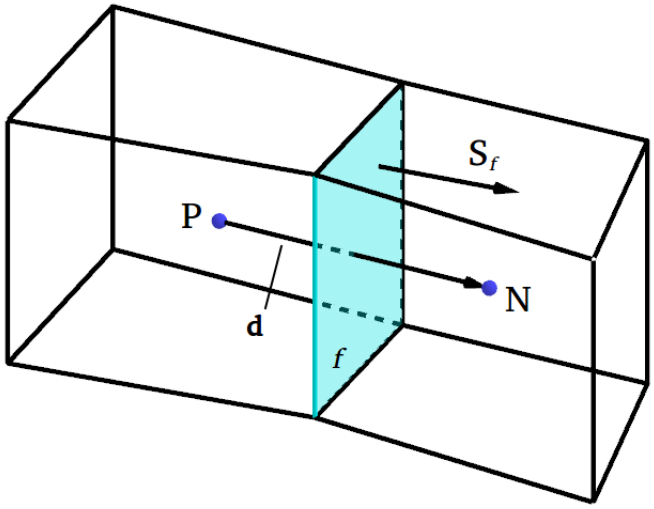


Figure 2.6: Control volume V_P with centroid P which is bounded by a set of flat faces, with face f shared with neighbour V_N with centroid N. $\mathbf{S}_f$ points out of owner cell.

variables share the same CVs. An example of a CV is show in Fig. 2.6, with a point P at the centroid of each CV such that it satisfies

$$V_P \mathbf{x}_P = \int_{V_{M_i}} \mathbf{x}\, dV_M \tag{2.39}$$

where $\mathbf{x}$ is the position of a point inside domain V_M and $\mathbf{x}_P$ is the location of the cell centroid. The CV is bounded by a set of flat faces and each face is shared with one neighbour CV, of cell centroid N. The cell faces are categorised as internal faces (those that are shared between two control volumes) and boundary faces (those that coincide with the domain boundaries). $\mathbf{S}_f$, the face area vector is constructed for each of the cell face such that it always points outwards of the CV with the lower label, is normal to the cell face, and whose magnitude represents the cell face area. Hereafter we define the cell with the lower label as the face 'owner' – its label is stored in the 'owner' array. The label of the other cell is now the 'neighbour', stored in the 'neighbour' array. For the boundary faces, their area vectors point toward the *outside* of the computational

domain and they are owned by the adjacent cells.

In Fig. 2.6, the owner cell centre is marked P and the neighbour cell centre marked N, as the face area vector $\mathbf{S}_f$ points outwards from the owner cell. All faces of the CV will now be denoted as f, which also represents the face centroid. The unit vector $\mathbf{n}$, which is normal to the face, is simply defined as $\mathbf{n} = \frac{\mathbf{S}}{|\mathbf{S}|}$ and we have $\mathbf{d}$ which denotes the vector between P and N i.e. $\mathbf{d} = \mathbf{x}_N - \mathbf{x}_P$. A mesh is said to be orthogonal when $\mathbf{d}$ is parallel to $\mathbf{S}_f$. OpenFOAM has no restrictions regarding the number of faces bounding each cell, but it is required that each cell be convex. This allows freedom in constructing 'unstructured' meshes which is useful when the spatial domain is complex.

Now consider the standard transport equation for a quantity γ,

$$\underbrace{\frac{\partial \rho\gamma}{\partial t}}_{\text{time derivative}} + \underbrace{\nabla \cdot (\rho \mathbf{u}\gamma)}_{\text{convective term}} = 0. \tag{2.40}$$

The accuracy of its discretisation will depend on the variation of γ in space and time. It assumed to be linear (Fig. 2.7) in both, as;

$$\gamma(\mathbf{x}) = \gamma_P + (\mathbf{x} - \mathbf{x}_P).(\nabla\gamma)_P, \tag{2.41}$$

$$\gamma(t + \Delta t) = \gamma^t + \Delta t\big(\frac{\partial \gamma}{\partial t}\big)^t, \tag{2.42}$$

where $\gamma_P = \gamma(\mathbf{x}_P)$, and $\gamma^t = \gamma(t)$. In the finite volume method, Eq.(2.40) needs to be satisfied over the control volume V_P around point P in the integral form;

$$\int_t^{t+\Delta t} \Big[\frac{\partial}{\partial t}\int_{V_P} \rho\gamma \; dV + \int_{V_P} \nabla \cdot (\rho \mathbf{u}\gamma) \; dV\Big] dt = 0 \tag{2.43}$$

2.3.2 Spatial discretisation

To discretise the spatial terms, the following generalised Gauss' theorem identities are used;

$$\int_V \nabla \cdot \mathbf{a} \; dV = \oint_{\partial V} \mathbf{a} \cdot d\mathbf{S} \tag{2.44}$$

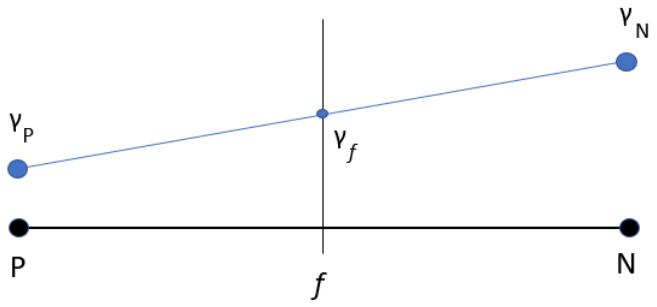


Figure 2.7: Linear variation of γ between points P and N

$$\int_V \nabla\gamma \, dV = \oint_{\partial V} d\mathbf{S} \, \gamma \tag{2.45}$$

where dV is the closed surface bounding volume V and $d\mathbf{S}$ is an infinitesimal surface element with an outward pointing normal, and $\mathbf{a}$ is a generic vector. Recalling Eq. (2.41), it follows that

$$\begin{aligned}\int_{V_P} \gamma(\mathbf{x}) \, dV &\cong \int_{V_P} \left[\gamma_P + (\mathbf{x} - \mathbf{x}_P) \cdot (\vec{\nabla}\gamma)_P\right] dV \\ &= \gamma_P \int_{V_P} dV + \int_{V_P} [(\mathbf{x} - \mathbf{x}_P) \, dV] \cdot (\vec{\nabla}\gamma)_P \\ &= \gamma_P V_P,\end{aligned} \tag{2.46}$$

as, from the definition of the centroid in Eq. (2.39),

$$\int_{V_P} [(\mathbf{x} - \mathbf{x}_P) \, dV] = 0. \tag{2.47}$$

Since it is assumed the variation of the transported property γ is linear, this leads to this expression for the face integral

$$\begin{aligned}\int_f \mathbf{a} \, d\mathbf{S} &= \mathbf{a}_f \cdot \int_f d\mathbf{S} + \left[\int_f (\mathbf{x} - \mathbf{x}_f) \, d\mathbf{S}\right] : (\nabla\mathbf{a})_f \\ &= \mathbf{a}_f \cdot \mathbf{S}_f.\end{aligned} \tag{2.48}$$

The terms with the divergence operator can be described in terms of the sum of integ-

rals over faces using the identity in Eq. (2.44):

$$\begin{aligned}\int_V \nabla \cdot \mathbf{a}\, dV &= \int_f \mathbf{a} \cdot d\mathbf{S} \\ (\nabla \cdot \mathbf{a})V_P &= \sum_j \int_f \mathbf{a} \cdot d\mathbf{S} = \sum_j \Big(\mathbf{a}_f \cdot \mathbf{S}_f\Big)_j \\ \nabla \cdot \mathbf{a} &= \frac{1}{V_P} \sum_j \Big(\mathbf{a}_f \cdot \mathbf{S}_f\Big)_j\end{aligned} \tag{2.49}$$

where $\mathbf{a}_f$ is the value of $\mathbf{a}$ at the face centroid and $\mathbf{S}_f$ is the face area vector for the same face. Recalling from Fig. 2.6, $\mathbf{S}_f$ points outwards of the owner P and into neighbour N. This needs to be taken into account in Eq. 2.49, so the sum over faces defined in terms of owner and neighbour faces becomes

$$\sum_f \mathbf{S} \cdot \mathbf{a}_f = \sum_{owner} \mathbf{S}_f \cdot \mathbf{a}_f - \sum_{neighbour} \mathbf{S}_f \cdot \mathbf{a}_f \tag{2.50}$$

The convective term in Eq. (2.40) can then be discretised as:

$$\begin{aligned}\int_{V_P} \nabla \cdot (\rho \mathbf{u} \phi)\, dV &= \sum_f \mathbf{S} \cdot (\rho \mathbf{u}\phi)_f \\ &= \sum_f \mathbf{S} \cdot (\rho \mathbf{u})_f \phi_f \\ &= \sum_f F \phi_f\end{aligned} \tag{2.51}$$

where F in Eq. 2.51 is the mass flux through the face

$$F = \mathbf{S}_f \cdot (\rho \mathbf{u})_f. \tag{2.52}$$

Gradient terms in the momentum equation are discretised using Eq. (2.45) as

$$\int_V \nabla \gamma\, dV = \oint_{\partial V} d\mathbf{S}\, \gamma = \sum_f \mathbf{S}_f \gamma_f. \tag{2.53}$$

2.3.3 Time discretisation

In the previous section, the discretisation of the spatial terms have been described; the surface and volume integrals are transformed into discrete sums and expressions

containing the face values of variables as a function of cell values. Consider again Eq. (2.43), which is the integral form of the transport equation;

$$\int_t^{t+\Delta t} \Big[\frac{\partial}{\partial t}\int_{V_P} \rho\gamma \; dV + \int_{V_P} \nabla\cdot(\rho\mathbf{u}\gamma) \; dV\Big] dt = 0$$

Assuming control volumes are constant in time, this can be written in its semi-discretised form;

$$\int_t^{t+\Delta t} \Big[\Big(\frac{\partial \rho\gamma}{\partial t}\Big)_P V_P + \sum_f F\gamma_f\Big] dt = 0 \tag{2.54}$$

With the variation of the function in time as in Eq. (2.42), the time integrals and derivatives are found directly as follows;

$$\Big(\frac{\partial \rho\gamma}{\partial t}\Big)_P = \frac{\rho_P^n\gamma_P^n - \rho_P^0\gamma_P^0}{\Delta t} \tag{2.55}$$

$$\int_t^{t+\Delta t} \gamma(t) \; dt = \frac{1}{2}(\gamma^0 + \gamma^n) \; \Delta t \tag{2.56}$$

where γ^n indicate the new time value, and γ^0 the old time value.

The temporal terms, where applicable, are discretised using the Euler implicit scheme. Although only first-order accurate, it is unconditionally stable.

2.4 Multi-moment methods for solving hyperbolic equations

The study of hyperbolic partial differential equations (PDE) are of substantial interest due to their prevalence in conservation laws. The most commonly used example of a hyperbolic PDE is the 1D advection (or transport) equation;

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0 \tag{2.57}$$

where f is a conserved scalar quantity and u is the velocity. Eq. (2.57) simply describes the time-dependent shift of f along x at velocity u. At any time $t > t_0$, the

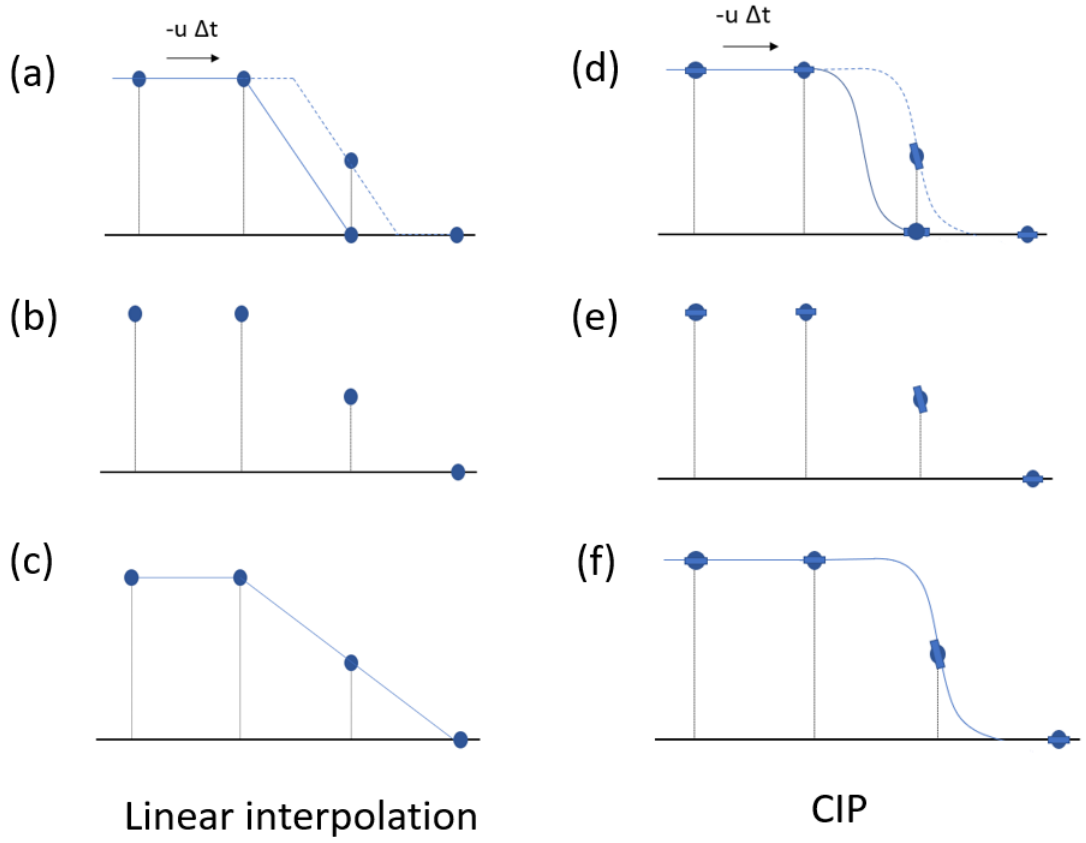


Figure 2.8: Reproduced from Figure 1 in [3], demonstrating the concept of the CIP method. The solid lines are the initial profile, with the dashed lines denoting the exact solution of the profile after advected by $-u\Delta t$, where u is the advection velocity and Δt is the time step. The profile is lost if using linear interpolation as in (a)-(c). Using the CIP method where the spatial derivative is also propagated, the profile in the grid can be reconstructed to a higher order of accuracy.

solution can be represented as a function of the state at time t_0;

$$f(x,t) = f(x - ut, 0). \tag{2.58}$$

This is termed an 'initial value problem', as the profile at any time $t > t_0$ can be determined uniquely if $t = t_0$ is known.

The exact solution is therefore a simple translational profile. However, in any numerical simulation, a discretisation process is inevitable. Since the grid resolution can only

be finite, some information regarding the profile will be lost in the discretisation process; profile information can only be stored at the grid points, hence any information between the grid points would be irretrievable as seen in Fig.2.8 (a)-(c) where a linear interpolation scheme is used. However, indiscriminately using high-order interpolation can cause oscillation where there exists discontinuities, even though they are desirable in smooth regions.

A large number of schemes have been proposed by various researchers to maintain the solution profile without causing numerical oscillation, such as the (Interpolated Differential Operator) IDO scheme [79], the Essentially Non-Oscillatory (ENO) schemes [80], a monotone cubic Hermite interpolation [81], and the Weighted Essentially Non-Oscillatory (WENO) schemes [82] [83].

Takewaki et al. [84] proposed the Cubic-Interpolated Propagation (CIP) method which is able to capture some sub-grid information by using cell gradient as well as cell boundary data. The usage of two or more different types of constraints for approximating the solution is termed the multi-moment approach. With the profile gradient as an additional constraint, the initial shape of the profile can be better maintained (Fig. 2.8 (d)-(f)) and an interpolation function of higher accuracy can be achieved with shorter computational stencils. Compact stencils are generally more desirable for the treatment of discontinuities as avoiding interpolating across discontinuities can help reduce numerical oscillation.

An extension of the CIP scheme is proposed by Yabe et al. [3], called the Constrained Interpolation Profile Conservative Semi-Lagrangian (CIP-CSL) family of methods [85] [86] [87] [88]. In the CIP-CSL method, mass conservation is guaranteed unlike the CIP method, as the spatial profile is constructed to satisfy an additional constraint: a cell-integrated value. In order to explain the CIP-CSL family of methods, consider the following one-dimensional conservation equation

$$\frac{\partial f}{\partial t} + \frac{\partial (uf)}{\partial x} = 0 \tag{2.59}$$

where f is the scalar property being transported. If the velocity u along x is constant,

this simply produces a translational motion. For velocity $u < 0$, one can approximate the profile inside the upwind cell as (if using a quadratic interpolation function);

$$\Phi_i^n(x) = C_{2,i}(x - x_i)^2 + C_{1,i}(x - x_i) + C_{0,i}, \tag{2.60}$$

where $C_{b,i}, b = 0, 1, 2$ are the coefficients unique for the specific type of interpolation. To obtain the profile at time step $n+1$, the profile is shifted by $-u\Delta t$. Towards this end, this interpolation function (Eq. 2.60) is built to best approximate the profile, using the cell average value and also the cell boundary values. It also has the advantage of being a compact high-order scheme, where a high-order polynomial can be constructed from the information contained only within a single cell. The usage of the cell average value as a constraint earns the CIP-CSL schemes conservative properties, which the CIP schemes lack using only cell boundary values and cell gradient values as constraints.

The following subsections describe two CIP-CSL schemes on which the contribution in this work is based; CIP-CSL2 [3] and CIP-CSL3 [89]. CIP-CSL2 and CIP-CSL3 differ in that different constraints are used to build the interpolation function Eq. (2.60).

2.4.1 CIP-CSL2

The CIP-CSL2 scheme involves three moments which are two cell boundary moments ($f_{i-1/2}, f_{i+1/2}$) and a cell average value $\overline{f}_i$. These moments are used to construct a second-order polynomial interpolation function

$$\Phi_i^{CSL2}(x) = C_{2,i}^{CSL2}(x - x_{i-\frac{1}{2}})^2 + C_{1,i}^{CSL2}(x - x_{i-\frac{1}{2}}) + f_{i-\frac{1}{2}} \tag{2.61}$$

for $u_{i-\frac{1}{2}} < 0$ with the constraints being

$$f_{i+1/2} = \Phi_i^{CSL2}(x_{i+1/2}), \tag{2.62}$$

$$\bar{f}_i = \frac{\int_{x_{i+1/2}}^{x_{i-1/2}} \Phi_i^{CSL2}(x)dx}{\Delta x}, \tag{2.63}$$

Using Equations (2.62) and (2.63), we obtain the following coefficients for the poly-

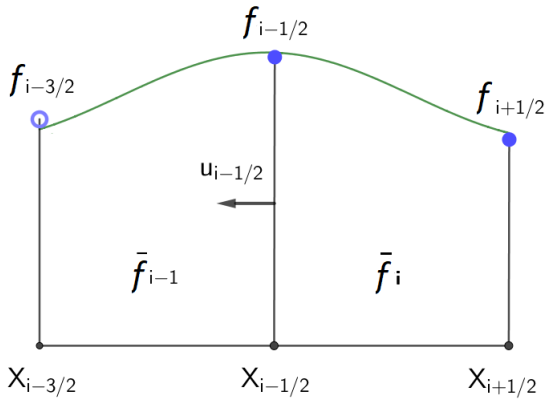


Figure 2.9: The moments used to build $\Phi_i^{CSL2}(x)$: $f_{i-1/2}, f_{i+1/2}, \overline{f}_i$

nomial

$$C_{1,i}^{CSL2} = \frac{1}{\Delta x}(6\bar{f}_i - 4f_{i-1/2} - 2f_{i+1/2}), \tag{2.64}$$

$$C_{2,i}^{CSL2} = \frac{1}{\Delta x^2}(-6\bar{f}_i + 3f_{i-1/2} + 3f_{i+1/2}). \tag{2.65}$$

Using the interpolation function $\Phi_i^{CSL2}(x)$ with the coefficients (2.64) and (2.65), the boundary value $f_{i-1/2}$ and the cell average $\bar{f}_i$ is updated by a third-order TVD Runge-Kutta (RK) formulation [90, 91] which is based on the CSL formulation solving the initial value problem as follows;

$$\frac{\partial X}{\partial t} = -u(X,t), \tag{2.66}$$

$$X_0 = x_{i-1/2}.$$

The third-order TVD Runge-Kutta method is as follows,

$$X_1 = X_0 - u(X_0, t_0)\Delta t, \tag{2.67}$$

$$X_2 = \frac{3}{4}X_0 + \frac{1}{4}X_1 - \frac{1}{4}u(X_1, t_1)\Delta t, \tag{2.68}$$

$$X_3 = \frac{1}{3}X_0 + \frac{2}{3}X_2 - \frac{2}{3}u(X_2, t_2)\Delta t. \tag{2.69}$$

Using the semi-Lagrangian scheme, $f_{i-1/2}$ at each RK time step can be obtained as;

$$f_{i-1/2}^{<k>} = \begin{cases} \Phi_{i-1}^{CSL2}(X_k) & \text{if} \quad X_k - X_0 \leq 0 \\ \Phi_{i}^{CSL2}(X_k) & \text{if} \quad X_k - X_0 > 0, \end{cases} \tag{2.70}$$

where k is the RK time step. The boundary value $f_{i-1/2}$ is updated for the new time step by solving the conservation equation in its differential form

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = -f\frac{\partial u}{\partial x}, \tag{2.71}$$

by

$$f_{i-1/2}^{n+1} = f_{i-1/2}^{<3>} - \frac{f_{i-1/2}^{<0>} + f_{i-1/2}^{<1>} + 4f_{i-1/2}^{<2>}}{6}\frac{\partial u}{\partial x}(X_0)\Delta t. \tag{2.72}$$

Meanwhile the cell average value $\bar{f}_i$ is updated for the new time step using a finite volume formulation as;

$$\bar{f}_i^{n+1} = \bar{f}_i^{n} - \frac{F_{i+1/2} - F_{i-1/2}}{\Delta x}, \tag{2.73}$$

where

$$F_{i-1/2} = \frac{f_{i-1/2}^{<0>} + f_{i-1/2}^{<1>} + 4f_{i-1/2}^{<2>}}{6}u(X_0). \tag{2.74}$$

2.4.2 CIP-CSL3

The CIP-CSL3 scheme uses the same three constraints as CSL2 ($\bar{f}_i, f_{i-1/2}, f_{i+1/2}$), with the addition of a slope (f_i') at the cell centre of the upwind cell. This produces the following third-order polynomial interpolation function for $u_{i-\frac{1}{2}} < 0$

$$\Phi_i^{CSL3}(x) = C_{3,i}^{CSL3}(x - x_{i-\frac{1}{2}})^3 + C_{2,i}^{CSL3}(x - x_{i-\frac{1}{2}})^2 + C_{1,i}^{CSL3}(x - x_{i-\frac{1}{2}}) + f_{i-\frac{1}{2}}. \tag{2.75}$$

The coefficients are;

$$C_{3,i}^{CSL3} = \frac{1}{\Delta x^3}(-4f_{i-1/2} + 4f_{i+1/2} - 4f_i'\Delta x), \tag{2.76}$$

$$C_{2,i}^{CSL3} = \frac{1}{\Delta x^2}(9f_{i-1/2} - 6\bar{f}_i - 3f_{i+1/2} + 6f_i'\Delta x), \tag{2.77}$$

$$C_{1,i}^{CSL3} = \frac{1}{\Delta x}(-6f_{i-1/2} + 6\bar{f}_i - 2f_i'\Delta x). \tag{2.78}$$

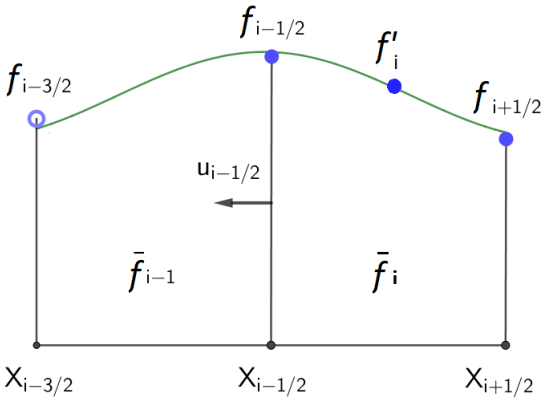


Figure 2.10: The moments used to build $\Phi_i^{CSL3}(x)$; $\bar{f}_i, f_{i-1/2}, f_{i+1/2}, f_i'$

Several formulations have been proposed to calculate the slope f_i' [89]. CSL3CW is one such formulation which is less oscillatory, and here it is estimated as

$$f_i' = \begin{cases} \min(|\hat{f}_{i+1} - \hat{f}_{i-1}|, 2|\hat{f}_{i+1} - \hat{f}_i|, 2|\hat{f}_i - \hat{f}_{i-1}|)* & \text{if } (\hat{f}_{i+1} - \hat{f}_i)(\hat{f}_i - \hat{f}_{i-1}) > 0 \\ \text{sgn}(\hat{f}_{i+1} - \hat{f}_{i-1})/\Delta x & \\ 0 & \text{otherwise,} \end{cases} \tag{2.79}$$

where

$$\hat{f}_i = \frac{3}{2}\bar{f}_i - \frac{1}{4}(f_{i+1/2} + f_{i-1/2}). \tag{2.80}$$

Another CSL3 formulation designated CSL3HYMAN [81] has f_i' estimated as

$$f_i' = \frac{\hat{f}_{i+2} + 8\hat{f}_{i+1} - 8\hat{f}_{i-1} - \hat{f}_{i-2}}{12\Delta x}. \tag{2.81}$$

CSL3HYMAN is 4th-order accurate for smooth solutions but suffers from oscillations at discontinuities.

2.5 Summary

The review was begun with a study of some existing interface capturing schemes; the Volume of Fluid method and the Level-Set method. To overcome the drawbacks of each of these methods, numerous researchers have proposed combining them, the foremost of which being the CLSVOF method. The open-source CFD code repository OpenFOAM implemented a VOF-based interface capturing approach for two-phase incompressible flows referred to as $interFoam$ where the interface is not explicitly tracked; instead an artificial compression term is used at the interface. The $interFoam$ scheme often exhibits interface smearing in simulations. In 2016, a contribution is made for OpenFOAM called the $interFlow$ solver by Roenby et al. [2] where an explicit reconstruction is performed to represent the fluid interface. The fluid interface in simulations obtained by $interFlow$ is very sharp and non-diffusive. However, given that $interFlow$ is originally developed for the simulation of ocean waves, it does not perform as well for surface tension dominant flows. This presents an opportunity that is addressed in this work: could the robust $interFlow$ method of interface capture be extended and coupled with a Level-Set method to harness $interFlow$'s sharp interface representation and the Level-Set method's capability in handling surface tension dominant flows? This is addressed in Chapters 4-5.

The second section of this review was concerning a family of hyperbolic equation solver called the CIP-CSL method. Based on a multi-moment implementation, the CIP-CSL schemes boast a high order of accuracy without resorting to large interpolation stencils. The CIP-CSL2 method uses a quadratic polynomial to approximate a profile with a cell average value used as one of the constraints, and the CIP-CSL3 method uses a cell gradient value in addition to cell average value to build a cubic polynomial. However, CSL2 is not completely oscillationless near discontinuities. CSL3, while oscillationless, is too smooth and cannot maintain the sharpness of the profile. To capture a discontinuity perfectly, a scheme must be able to resolve the steep change

in profile but without giving way to numerical oscillation. With this basis in mind, in this work a new CIP-CSL scheme is proposed to address these issues, that is, how can a CIP-CSL scheme be built such that sharp profiles can be preserved but numerical oscillations can be suppressed? This is addressed in Chapter 3.

Chapter 3

Hyperbolic Equation Solver based on the Multi-Moment Method

In this chapter, a new hyperbolic equation solver from the family of CIP-CSL methods is proposed. The CIP-CSL2 [3] and CIP-CSL3 [89] methods have been explained in Section 2.4 where the former uses three moments in the upwind cell to build a quadratic interpolation function, and the latter uses four moments to build a cubic interpolation function. Using the more recent variant of the CIP-CSL3 method, CIP-CSL3D [92], another complementary scheme is proposed, called CIP-CSL3U [93]. These two methods are then combined to achieve two new high-order schemes that do not suffer from excessive oscillation near discontinuities.

3.1 CIP-CSL3D

The CIP-CSL3D [92] method uses three moments in the upwind cell (a cell average moment $\bar{f}_i$ and two cell boundary values $f_{i-\frac{1}{2}}$ and $f_{i+\frac{1}{2}}$) and one moment in the downwind cell centre ($\hat{f}_{i-1}$). $\hat{f}_{i-1}$ is interpolated from the downwind cell's boundary values and cell integrated average. These moments yield a cubic interpolation function as shown below for $u_{i-\frac{1}{2}} < 0$

$$\Phi_i^{CSL3D}(x) = C_{3,i}^{CSL3D}(x - x_{i-\frac{1}{2}})^3 + C_{2,i}^{CSL3D}(x - x_{i-\frac{1}{2}})^2 + C_{1,i}^{CSL3D}(x - x_{i-\frac{1}{2}}) + f_{i-\frac{1}{2}} \quad (3.1)$$

The constraints are

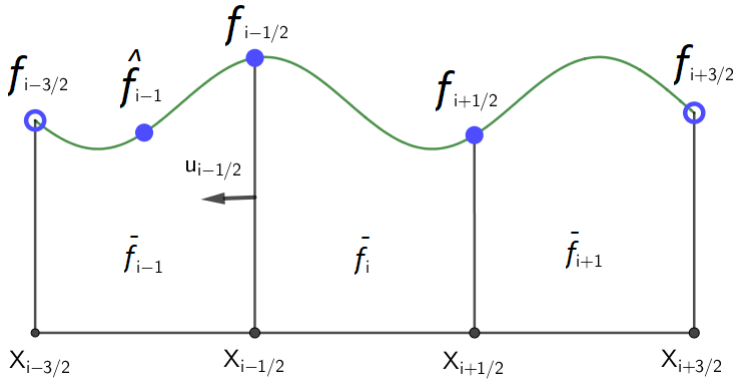


Figure 3.1: The moments used to build $\Phi_i^{CSL3D}(x)$

$$f_{i-1/2} = \Phi_i^{CSL3D}(x_{i-1/2}), \tag{3.2}$$

$$\bar{f}_i = \frac{\int_{x_{i-1/2}}^{x_{i+1/2}} \Phi_i^{CSL3D}(x)dx}{\Delta x}, \tag{3.3}$$

$$\hat{f}_{i-1} = \Phi_i^{CSL3D}(x_{i-1}), \tag{3.4}$$

where $\hat{f}_{i-1}$ is calculated as;

$$\hat{f}_{i-1} = \frac{3}{2}\bar{f}_{i-1} - \frac{1}{4}(f_{i-3/2} + f_{i-1/2}). \tag{3.5}$$

The coefficients are then obtained as follows

$$C_{3,i}^{CSL3D} = \frac{1}{3\Delta x^3}(-4\hat{f}_{i-1} + 15f_{i-1/2} - 18\bar{f}_i + 7f_{i+1/2}), \tag{3.6}$$

$$C_{2,i}^{CSL3D} = \frac{1}{2\Delta x^2}(-4\hat{f}_{i-1} + 9f_{i-1/2} - 6\bar{f}_i + f_{i+1/2}), \tag{3.7}$$

$$C_{1,i}^{CSL3D} = -\frac{1}{6\Delta x}(4\hat{f}_{i-1} + 9f_{i-1/2} - 18\bar{f}_i + 5f_{i+1/2}). \tag{3.8}$$

Using the interpolation function $\Phi_i^{CSL3D}(x)$, the boundary value $f_{i-1/2}$, and cell average $\bar{f}_i$ are updated using a third-order TVD Runge-Kutta formulation as detailed in Section (2.4.1).

3.2 CIP-CSL3U

The CIP-CSL3U interpolation scheme is proposed, which is another variant of the CIP-CSL3 method. It is complementary to CSL3D; while CSL3D uses three constraints in the upwind cell and one constraint in the downwind cell, CSL3U has all four moments in the upwind cell. For $u_{i-1/2} < 0$, the interpolation function is

$$\Phi_i^{CSL3U}(x) = C_{3,i}^{CSL3U}(x-x_{i-1/2})^3+C_{2,i}^{CSL3U}(x-x_{i-1/2})^2+C_{1,i}^{CSL3U}(x-x_{i-1/2})+f_{i-1/2}. \tag{3.9}$$

obtained using the following constraints

$$f_{i-1/2} = \Phi_i^{CSL3U}(x_{i-1/2}) \tag{3.10}$$

$$\bar{f}_i = \frac{\int_{x_{i-1/2}}^{x_{i+1/2}} \Phi_i^{CSL3U}(x)dx}{\Delta x}, \tag{3.11}$$

$$f_{i+1/2} = \Phi_i^{CSL3U}(x_{i+1/2}), \tag{3.12}$$

$$\hat{f}_{i+1} = \Phi_i^{CSL3U}(x_{i+1}). \tag{3.13}$$

The coefficients are then

$$C_{3,i}^{CSL3U} = \frac{1}{3\Delta x^3}(-7f_{i-1/2} + 18\bar{f}_i - 15f_{i+1/2} + 4\hat{f}_{i+1}), \tag{3.14}$$

$$C_{2,i}^{CSL3U} = \frac{1}{2\Delta x^2}(13f_{i-1/2} - 30\bar{f}_i + 21f_{i+1/2} - 4\hat{f}_{i+1}), \tag{3.15}$$

$$C_{1,i}^{CSL3U} = \frac{1}{6\Delta x}(-31f_{i-1/2} + 54\bar{f}_i - 27f_{i+1/2} + 4\hat{f}_{i+1}). \tag{3.16}$$

For $u_{i-1/2} \geq 0$, the interpolation function is then

$$\Phi_{i-1}^{CSL3U}(x) = C_{3,i-1}^{CSL3U}(x-x_{i-1/2})^3+C_{2,i-1}^{CSL3U}(x-x_{i-1/2})^2+C_{1,i-1}^{CSL3U}(x-x_{i-\frac{1}{2}})+f_{i-1/2} \tag{3.17}$$

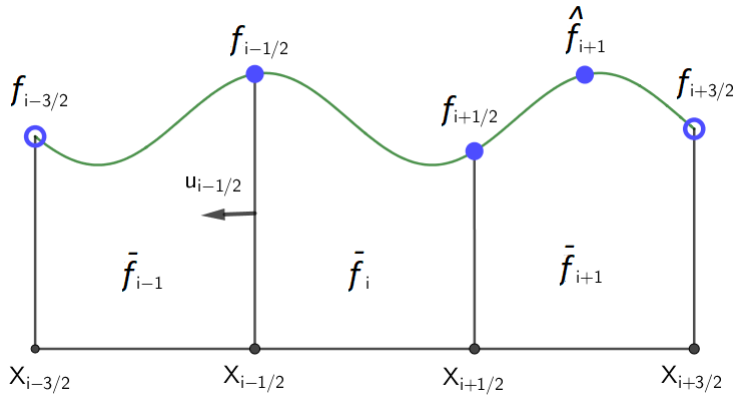


Figure 3.2: Moments used to build $\Phi_i^{CSL3U}(x)$

where the coefficients are

$$C_{3,i-1}^{CSL3U} = -\frac{1}{3\Delta x^3}(-7f_{i-1/2} + 18\bar{f}_{i-1} - 15f_{i-3/2} + 4\hat{f}_{i-2}), \tag{3.18}$$

$$C_{2,i-1}^{CSL3U} = \frac{1}{2\Delta x^2}(13f_{i-1/2} - 30\bar{f}_{i-1} + 21f_{i-3/2} - 4\hat{f}_{i-2}), \tag{3.19}$$

$$C_{1,i-1}^{CSL3U} = -\frac{1}{6\Delta x}(-31f_{i-1/2} + 54\bar{f}_{i-1} - 27f_{i-3/2} + 4\hat{f}_{i-2}). \tag{3.20}$$

As in CIP-CSL3D, the boundary value $f_{i-1/2}$, and cell average $\bar{f}_i$ are updated using a third-order TVD Runge-Kutta formulation as detailed in Section (2.4.1).

3.3 Fourier analysis

Fourier analysis [79] is conducted for the proposed CSL3U scheme and compared to the existing CSL3D. The results show the resolution of the spatial derivatives in wavenumber space. The spatial profile of $\Phi(x)$ is defined over the domain $[0, L]$ with a uniform grid spacing of Δx, and is decomposed into Fourier series as

$$\Phi(x) = \sum_k \Phi(k)\, e^{jwx/\Delta x}, \tag{3.21}$$

where $j = \sqrt{-1}$, and $w = 2\pi k \Delta x / L$ is the scaled wavenumber. As an example, the point value at $x_{i-1/2}$ is decomposed as

$$\Phi_{i-1/2} = \sum_k \Phi(k)\, e^{jwx_{i-1/2}/\Delta x}. \tag{3.22}$$

Using Eq.(3.22), we can then generalise for values $x_{i-1/2+m}$

$$\Phi_{i-1/2+m} = \Phi_{i-1/2}\, e^{jwm}. \tag{3.23}$$

Using this, the cell average Φ_i is decomposed as

$$\bar{\Phi}_i = \frac{1}{\Delta x} \int_0^{\Delta x} \Phi(x_{i-1/2} + x)\, dx = \Phi_{i-1/2}\, \frac{e^{jw} - 1}{jw} \tag{3.24}$$

which demonstrates the relationship between the point values and the cell average. In this study, the spatial derivatives of each scheme are examined around three points; x_i, $x_{i-1/2}$, and $x_{i+1/2}$. For example, to obtain the spatial derivative of CSL3U at $x_{i-1/2}$ in Fourier space, the boundary value $f_{i-1/2}$ is decomposed as given by Eq. (3.23) and the cell average value is decomposed as Eq. (3.24). The decomposed coefficients in $C_{1,i}^{CSL3U}$ (Eq. (3.16)) then corresponds to the first derivative $\Phi_x(w)$ in Eq. (3.31).

The following equations are the formulations of Fourier analysis for CSL3U and CSL3D around x_i, $x_{i-1/2}$, and $x_{i+1/2}$.

3.3.0.1 The spatial derivatives of CSL3U at x_i in Fourier space

$$\begin{aligned}\Phi_x(w) = \frac{1}{12}\left(\cos(\frac{3w}{2}) + 23\cos(\frac{w}{2}) - \frac{30}{w}\sin(\frac{w}{2}) - \frac{6}{w}\sin(\frac{3w}{2})\right) \\ + \left(\sin(\frac{3w}{2}) + 33\sin(\frac{w}{2}) + \frac{6}{w}\cos(\frac{3w}{2}) - \frac{6}{w}\cos(\frac{w}{2})\right) j,\end{aligned} \tag{3.25}$$

$$\Phi_{xx}(w) = \left(12\cos(\frac{w}{2}) - \frac{24}{w}\sin(\frac{w}{2})\right), \tag{3.26}$$

$$\begin{aligned}\Phi_{xxx}(w) = \left(-\cos(\frac{3w}{2}) - 23\cos(\frac{w}{2}) + \frac{30}{w}\sin(\frac{w}{2}) + \frac{6}{w}\sin(\frac{3w}{2})\right) \\ - \left(\sin(\frac{3w}{2}) + 9\sin(\frac{w}{2}) + \frac{6}{w}\cos(\frac{3w}{2}) - \frac{6}{w}\cos(\frac{w}{2})\right) j.\end{aligned} \tag{3.27}$$

3.3.0.2 The spatial derivatives of CSL3D at x_i in Fourier space

$$\Phi_x(w) = \frac{1}{12}\left(-\cos(\frac{3w}{2}) - 23\cos(\frac{w}{2}) + \frac{30}{w}\sin(\frac{w}{2}) + \frac{6}{w}\sin(\frac{3w}{2})\right) + \left(\sin(\frac{3w}{2}) + 33\sin(\frac{w}{2}) + \frac{6}{w}\cos(\frac{3w}{2}) - \frac{6}{w}\cos(\frac{w}{2})\right)j, \tag{3.28}$$

$$\Phi_{xx}(w) = \left(12\cos(\frac{w}{2}) - \frac{24}{w}\sin(\frac{w}{2})\right), \tag{3.29}$$

$$\Phi_{xxx}(w) = \left(\cos(\frac{3w}{2}) + 23\cos(\frac{w}{2}) - \frac{30}{w}\sin(\frac{w}{2}) - \frac{6}{w}\sin(\frac{3w}{2})\right) - \left(\sin(\frac{3w}{2}) + 9\sin(\frac{w}{2}) + \frac{6}{w}\cos(\frac{3w}{2}) - \frac{6}{w}\cos(\frac{w}{2})\right)j. \tag{3.30}$$

3.3.0.3 The spatial derivatives of CSL3U at $x_{i-1/2}$ in Fourier space

$$\Phi_x(w) = \left(-\frac{31}{6} - \frac{14}{3}\cos(w) + \frac{8}{w}\sin(w) + \frac{1}{w}\sin(2w) - \frac{1}{6}\cos(2w)\right) + \left(-\frac{14}{3}\sin(w) - \frac{8}{w}\cos(w) + \frac{9}{w} - \frac{1}{w}\cos(2w) - \frac{1}{6}\sin(2w)\right)j, \tag{3.31}$$

$$\Phi_{xx}(w) = \left(13 + 22\cos(w) - \frac{24}{w}\sin(w) - \frac{6}{w}\sin(2w) + \cos(2w)\right) + \left(22\sin(w) - \frac{30}{w} + \frac{24}{w}\cos(w) + \frac{6}{w}\cos(2w) + \sin(2w)\right)j, \tag{3.32}$$

$$\Phi_{xxx}(w) = \left(-14 - 32\cos(w) + \frac{24}{w}\sin(w) + \frac{12}{w}\sin(2w) - 2\cos(2w)\right) + \left(-32\sin(w) + \frac{36}{w} - \frac{24}{w}\cos(w) - \frac{12}{w}\cos(2w) - 2\sin(2w)\right)j. \tag{3.33}$$

3.3.0.4 The spatial derivatives of CSL3D at $x_{i-1/2}$ in Fourier space

$$\Phi_x(w) = \left(-\frac{2}{3}\cos(w) + \frac{2}{w}\sin(w) - \frac{4}{3}\right) + \left(-\sin(w) + \frac{4}{w} - \frac{4}{w}\cos(w)\right)j, \tag{3.34}$$

$$\Phi_{xx}(w) = -2\cos(w) + \frac{12}{w}\sin(w) - 10, \tag{3.35}$$

$$\Phi_{xxx}(w) = \left(16\cos(w) - \frac{48}{w}\sin(w) + 32\right) + \left(12\sin(w) + \frac{24}{w}\cos(w) - \frac{24}{w}\right)j \tag{3.36}$$

3.3.0.5 The spatial derivatives of CSL3U at $x_{x+1/2}$ in Fourier space

$$\Phi_x(w) = -\left(-\frac{2}{3}\cos(w) + \frac{2}{w}\sin(w) - \frac{4}{3}\right) + \left(-\sin(w) + \frac{4}{w} - \frac{4}{w}\cos(w)\right)j, \tag{3.37}$$

$$\Phi_{xx}(w) = -2\cos(w) + \frac{12}{w}\sin(w) - 10, \tag{3.38}$$

$$\Phi_{xxx}(w) = -\left(16\cos(w) - \frac{48}{w}\sin(w) + 32\right) + \left(12\sin(w) + \frac{24}{w}\cos(w) - \frac{24}{w}\right)j \tag{3.39}$$

3.3.0.6 The spatial derivatives of CSL3D at $x_{x+1/2}$ in Fourier space

$$\begin{aligned}\Phi_x(w) = &-\left(-\frac{31}{6} - \frac{14}{3}\cos(w) + \frac{8}{w}\sin(w) + \frac{1}{w}\sin(2w) - \frac{1}{6}\cos(2w)\right) \\ &+\left(-\frac{14}{3}\sin(w) - \frac{8}{w}\cos(w) + \frac{9}{w} - \frac{1}{w}\cos(2w) - \frac{1}{6}\sin(2w)\right)j,\end{aligned} \tag{3.40}$$

$$\begin{aligned}\Phi_{xx}(w) = &\left(13 + 22\cos(w) - \frac{24}{w}\sin(w) - \frac{6}{w}\sin(2w) + \cos(2w)\right) \\ &-\left(22\sin(w) - \frac{30}{w} + \frac{24}{w}\cos(w) + \frac{6}{w}\cos(2w) + \sin(2w)\right)j,\end{aligned} \tag{3.41}$$

$$\begin{aligned}\Phi_{xxx}(w) = &-\left(-14 - 32\cos(w) + \frac{24}{w}\sin(w) + \frac{12}{w}\sin(2w) - 2\cos(2w)\right) \\ &+\left(-32\sin(w) + \frac{36}{w} - \frac{24}{w}\cos(w) - \frac{12}{w}\cos(2w) - 2\sin(2w)\right)j.\end{aligned} \tag{3.42}$$

Figures 3.3, 3.4 and 3.5 show the results of the imaginary components of Fourier analysis of CSL3U with those of CSL3D at x_i, $x_{i-1/2}$ and $x_{i+1/2}$, respectively.

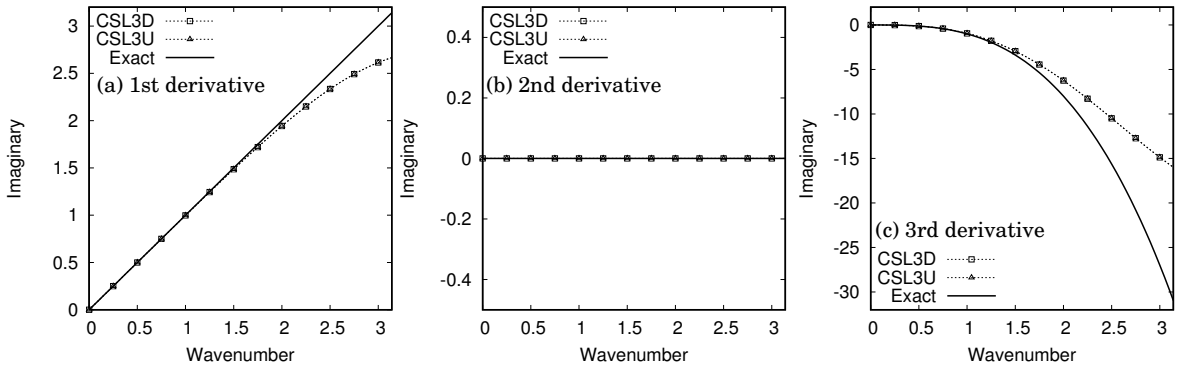


Figure 3.3: Spatial derivatives of CSL3D and CSL3U at the cell center x_i. (a), (b) and (c) show results of imaginary parts of first, second and third derivatives, respectively.

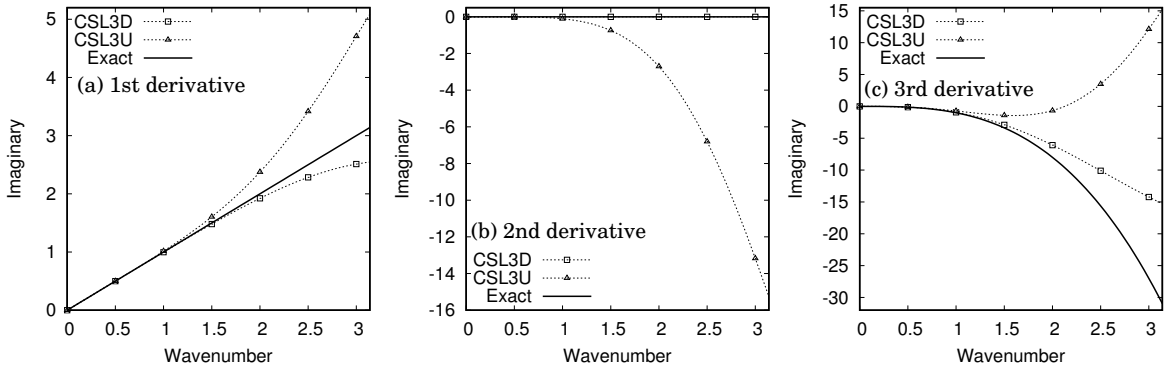


Figure 3.4: Spatial derivatives of CSL3D and CSL3U at a cell boundary $x_{i-1/2}$. (a), (b) and (c) show results of imaginary parts of first, second and third derivatives, respectively.

Consider Fig. 3.4, which shows the imaginary components (corresponding to advection speed) of the spatial derivatives of CSL3D and CSL3U at the cell boundary $x_{i-1/2}$. For all three derivatives, it is shown that CSL3D is superior to CSL3U in that it approximates the exact value more closely. This is due to the additional constraint for CSL3D ($\hat{f}_{i-1}$) being closer to the point $x_{i-1/2}$. However, the case is reversed when we examine their spatial derivatives at $x_{i+1/2}$; in this situation CSL3U is shown to be the more accurate. It should also be noted that the results at $x_{i+1/2}$ and $x_{i+1/2}$ are exactly sym-

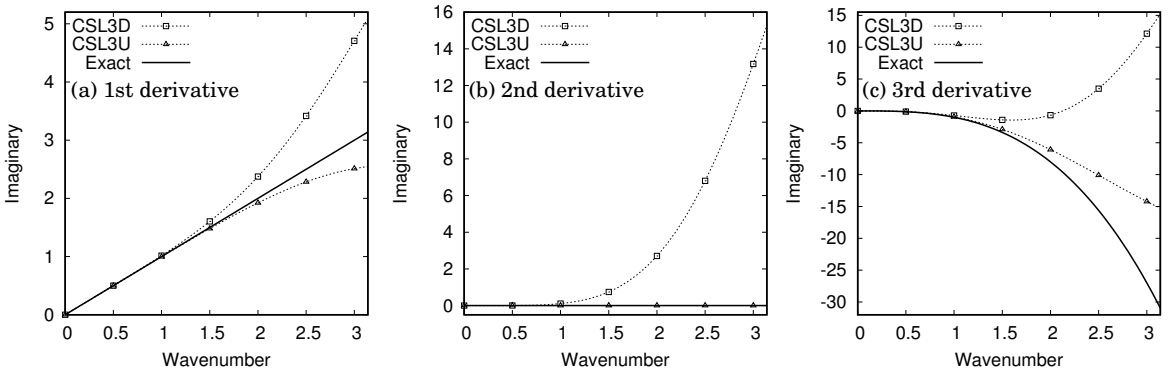


Figure 3.5: Spatial derivatives of CSL3D and CSL3U at a cell boundary $x_{i+1/2}$. (a), (b) and (c) show results of imaginary parts of first, second and third derivatives, respectively.

metrical, as further proved by Figure 3.3 showing that the spatial derivatives of CSL3D and CSL3U taken at the cell centre x_i are exactly equal.

This analysis suggests that CSL3D and CSL3U are good candidates for an ENO-like formulation due to their complementary nature. Where CSL3D produces more errors, we can switch to using CSL3U, and vice versa.

3.4 CSL3DU formulation

A new ENO-based scheme called CSL3DU is proposed. The ENO (essentially non-oscillatory) method was first introduced by Harten et al. [80] in 1987. It has been improved upon by many researchers over the years [94, 95, 96, 97] and also brought forth the WENO family of methods [83, 98, 99, 100, 101]. The keystone of ENO is the real-time selection of stencils during simulations. In order to avoid interpolating over discontinuities, the scheme selects an appropriate, locally smoother stencil over several candidates. To be able to do so properly, one must be able to identify discontinuities in the solution, which enables the correct stencil to be chosen. The following smoothness indicator to select between CSL3D and CSL3U is proposed, which is a modified

version of the indicator proposed by Zhang and Shu [102].

$$I_{CSL3D} = 4\int_{x-\frac{1}{8}}^{x+\frac{1}{8}} \Delta x\Big(\frac{\delta f^{CSL3D}(x)}{\delta x}\Big)^2 dx - \frac{7}{9}\int_{x-\frac{1}{8}}^{x+\frac{1}{8}} \Delta x^3\Big(\frac{\delta^2 f^{CSL3D}(x)}{\delta^2 x}\Big)^2 dx \quad (3.43)$$

$$I_{CSL3U} = 4\int_{x-\frac{1}{8}}^{x+\frac{1}{8}} \Delta x\Big(\frac{\delta f^{CSL3U}(x)}{\delta x}\Big)^2 dx - \frac{7}{9}\int_{x-\frac{1}{8}}^{x+\frac{1}{8}} \Delta x^3\Big(\frac{\delta^2 f^{CSL3U}(x)}{\delta^2 x}\Big)^2 dx \quad (3.44)$$

The smoothness indicator is used to calculate a combination of the first and second derivatives of the CSL3D and CSL3U local interpolation. A lower indicator value signals a smoother reconstruction, so the stencil with a lower indicator value is selected (see Figure 3.7). Figure 3.6 shows CSL3D and CSL3U being used to individually transport a square wave.

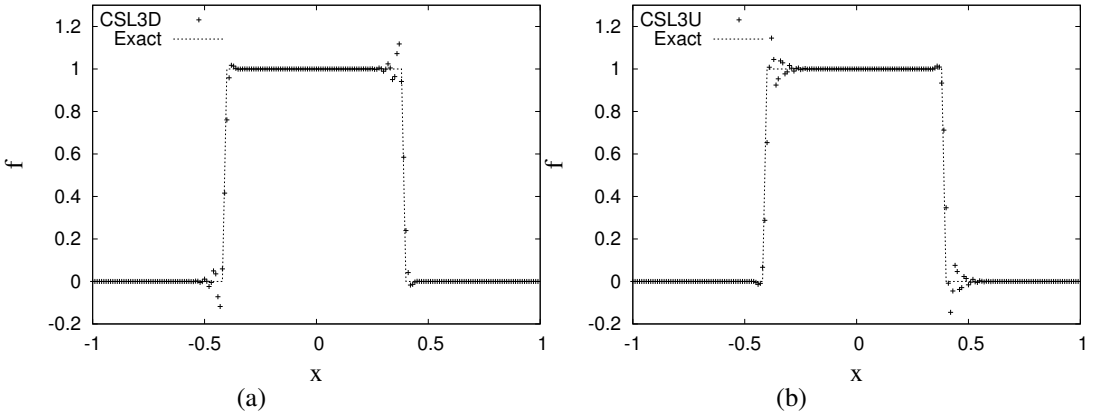


Figure 3.6: Numerical results of square wave propagation at 500 time steps (1 cycle) using $CFL = 0.4$ for (a) CSL3D and (b) CSL3U.

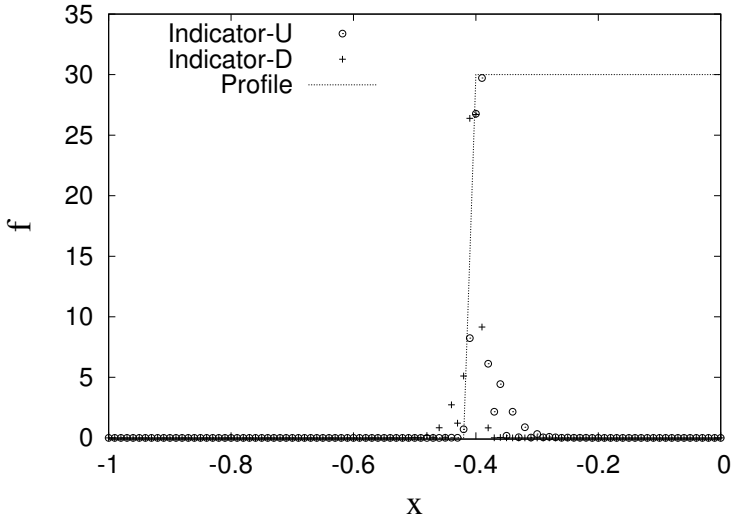


Figure 3.7: Distribution of the smoothness indicator as applied to CSL3D and CSL3U on the square wave as in Fig. 3.6, zoomed in to the region $-1 < x < 0$. The wave profile has been enlarged to $f = 30$ from $f = 1$ in order to better juxtapose with the indicator values.

Figure 3.6 demonstrates that the region immediately before a discontinuity is better approximated by CSL3U, whereas the region immediately after is better handled by CSL3D. Figure 3.7 shows how the indicator is valued at critical locations; it can be seen that the indicator for CSL3D has a lower value after a sharp jump, and CSL3U has a lower value before a sharp jump.

3.5 CSL3ENO formulation

A second smoothness indicator is proposed [93] based on the ratios of successive gradients. Consider the situation for $u_{i-1/2} < 0$. The smoothness of the interpolation functions of CSL3D and CSL3U are evaluated using the ratios of successive gradients $r_{i-1/2}$ and $r_{i+1/2}$, where

$$r_{i-1/2} = \text{sgn}\Big(\frac{\hat{f}_i - f_{i-1/2}}{f_{i-1/2} - \hat{f}_{i-1}}\Big)\max\Big(\frac{|\hat{f}_i - f_{i-1/2}|}{|f_{i-1/2} - \hat{f}_i|}, \frac{|f_{i-1/2} - \hat{f}_{i-1}|}{|\hat{f}_i - f_{i-1/2}|}\Big). \tag{3.45}$$

$$r_{i+1/2} = \mathrm{sgn}\Big(\frac{\hat{f}_{i+1} - f_{i+1/2}}{f_{i+1/2} - \hat{f}_i}\Big)\max\Big(\frac{|\hat{f}_{i+1} - f_{i+1/2}|}{|f_{i+1/2} - \hat{f}_i|}, \frac{|f_{i+1/2} - \hat{f}_i|}{|\hat{f}_{i+1} - f_{i+1/2}|}\Big). \tag{3.46}$$

For $u_{i-1/2} > 0$, the smoothness of CSL3D is evaluated by $r_{i-1/2}$, and for CSL3U by $r_{i-3/2}$.

The stencil is selected using the following algorithm:

1. If $r_{i-1/2}$ and $r_{i+1/2}$ have the same signs, select the larger value.
2. If $r_{i-1/2}$ and $r_{i+1/2}$ have different signs, select the one with the negative sign.

A negative $r_{i-1/2}$ or $r_{i+1/2}$ value indicates slopes in opposite directions. The natural instinct would be to avoid such stencils as gradients in opposite directions may indicate a discontinuous profile. However, it is demonstrated that CSL3D and CSL3U are already able to handle opposite slopes as demonstrated in Fig. 3.9. This may be a reason for the effectiveness of the CSL3ENO formulation. Although the exact mechanism of this selector is not well understood, results show that the CSL3ENO formulation worked very well in benchmark tests.

3.6 Results

The proposed methods are validated using various benchmark tests and compared with CIP-CSL3CW and CIP-CSL2, as well as CSL3D and CSL3U individually.

3.6.1 Sine wave propagation

The conservation equation is solved with the initial condition

$$f(x, 0) = \sin(2\pi x) \tag{3.47}$$

with the computation domain $[0:1]$ and velocity $u(x,0) = 1$, with periodic boundary conditions. Different grid sizes are used (N=50, 100, 200, 400, 800) and $\Delta t = 0.4\Delta x$ with $\Delta x = 1/N$. Errors are calculated as follows

$$L_1 = \frac{1}{N}\sum_{i=1}^{N} |f_i - f_{exact,i}|, \tag{3.48}$$

$$L_\infty = \max(|f_i - f_{exact,i}|). \tag{3.49}$$

and are shown in Table 3.1.

Test results indicate that CSL3D and CSL3U are both 4^{th} order accurate, compared to CSL2 which is 3^{rd} order accurate and CSL3CW which is only around 2^{nd} order accurate due to the usage of a slope limiter. It is seen that CSL3DU and CSL3ENO more or less maintained the 4^{th} order accuracy of its constituents CSL3D and CSL3U.

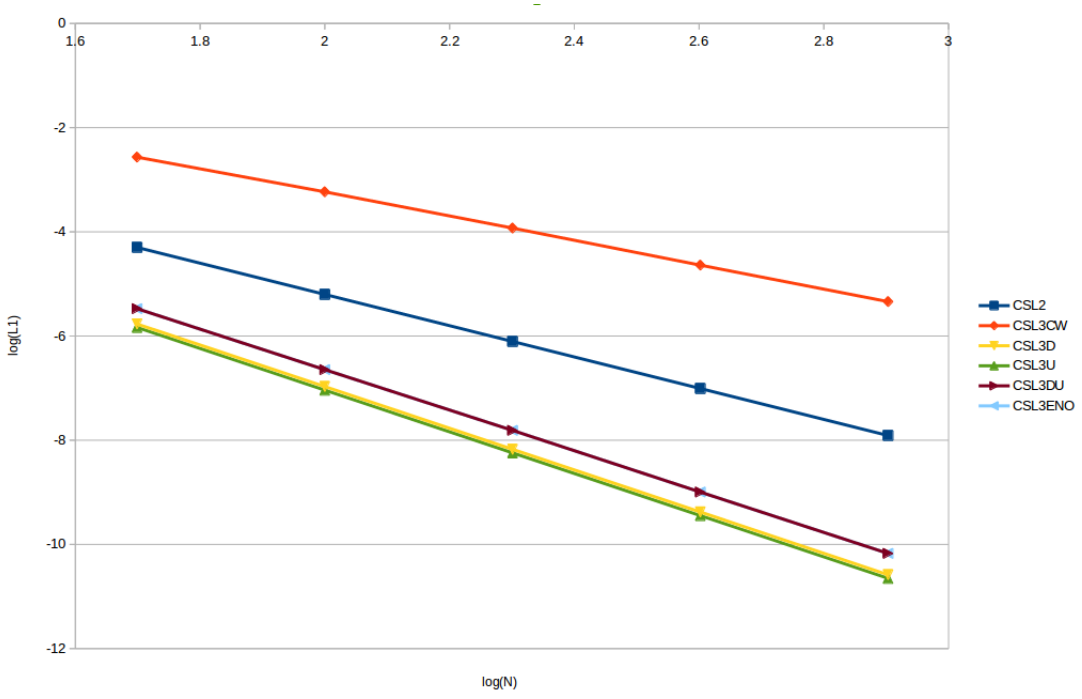


Figure 3.8: Comparison of L_1 error in the sine wave refinement test

Table 3.1: Errors in sine wave propagation at t=1.

Method	N	L_1 error	L_1 order	L_∞ error	L_∞ order
CSL2	50	5.02×10^{-5}	-	7.88×10^{-5}	-
	100	6.28×10^{-6}	3.00	9.86×10^{-6}	3.00
	200	7.85×10^{-7}	3.00	1.23×10^{-6}	3.00
	400	9.82×10^{-8}	3.00	1.54×10^{-7}	3.00
	800	1.23×10^{-8}	3.00	1.93×10^{-8}	3.00
CSL3CW	50	2.71×10^{-3}	-	1.53×10^{-2}	-
	100	5.85×10^{-4}	2.21	5.04×10^{-3}	1.60
	200	1.18×10^{-4}	2.31	1.80×10^{-3}	1.49
	400	2.29×10^{-5}	2.36	6.27×10^{-4}	1.52
	800	4.57×10^{-6}	2.33	2.14×10^{-4}	1.55
CSL3D	50	1.70×10^{-6}	-	2.68×10^{-6}	-
	100	1.07×10^{-7}	3.99	1.68×10^{-7}	4.00
	200	6.68×10^{-9}	4.00	1.05×10^{-8}	4.00
	400	4.18×10^{-10}	4.00	6.56×10^{-10}	4.00
	800	2.61×10^{-11}	4.00	4.10×10^{-11}	4.00
CSL3U	50	1.46×10^{-6}	-	2.29×10^{-6}	-
	100	9.13×10^{-8}	4.00	1.43×10^{-7}	4.00
	200	5.71×10^{-9}	4.00	8.97×10^{-9}	3.99
	400	3.57×10^{-10}	4.00	5.60×10^{-10}	4.00
	800	2.23×10^{-11}	4.00	3.50×10^{-10}	4.00
CSL3DU	50	3.35×10^{-6}	-	2.32×10^{-5}	-
	100	2.26×10^{-7}	3.89	2.47×10^{-6}	3.23
	200	1.54×10^{-8}	3.88	2.88×10^{-7}	3.10
	400	1.01×10^{-9}	3.93	3.32×10^{-8}	3.12
	800	6.70×10^{-11}	3.91	3.80×10^{-9}	3.13
CSL3ENO	50	3.35×10^{-6}	-	2.32×10^{-5}	-
	100	2.26×10^{-7}	3.89	2.47×10^{-6}	3.23
	200	1.54×10^{-8}	3.88	2.89×10^{-7}	3.10
	400	1.01×10^{-9}	3.93	3.33×10^{-8}	3.13
	800	6.70×10^{-11}	3.91	3.80×10^{-9}	3.13

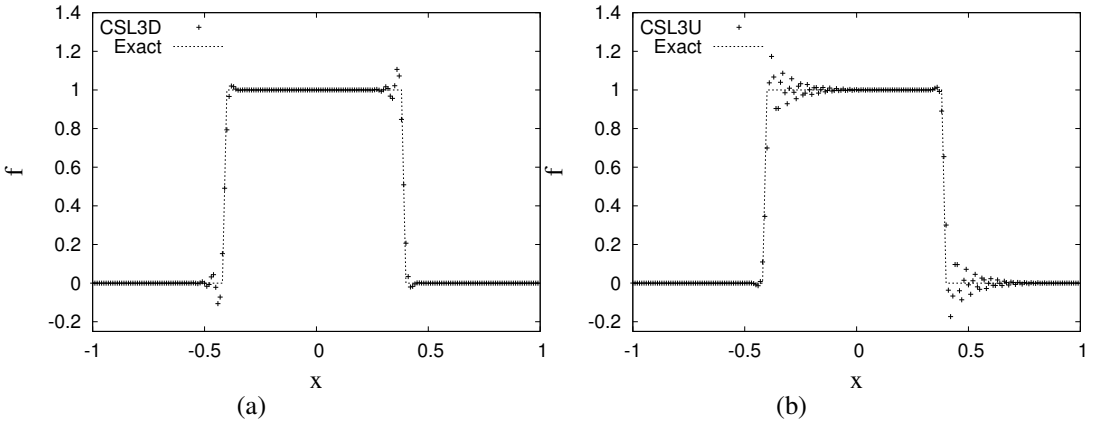


Figure 3.9: Numerical results of square wave propagation at $t = 1$ (500 time steps) with $CFL = 0.2$.

3.6.2 Square wave propagation

CSL3D, CSL3U, and CSL3DU are tested using a square wave as in Subsection 3.4 but with different CFL numbers. Mesh size $N = 200$ is used where $\Delta x = 2/N$ with domain $[-1 : 1]$ and periodic boundary conditions. The initial condition is set as

$$f(x,0) = \begin{cases} 1 & \text{if} \quad -0.4 \leqslant x \leqslant 0.4, \\ 0 & \text{otherwise} \end{cases} \tag{3.50}$$

and the test is run with $CFL = 0.2$ ($\Delta t = 0.2\Delta x$), $CFL = 0.5$ ($\Delta t = 0.5\Delta x$), and $CFL = 0.8$ ($\Delta t = 0.8\Delta x$). Results indicate that at $CFL = 0.2$, CSL3D performs better than CSL3U. At $CFL = 0.8$, CSL3U is seen to perform better, and at $CFL = 0.5$, CSL3D and CSL3U has symmetrical results with oscillations occurring at opposite sides of the discontinuity. The reason for this phenomenon is that at $CFL = 0.2$, the departure point is located closer to the additional moment of CSL3D ($\hat{f}_{i-1}$ in Fig. 3.1). Meanwhile when using $CFL = 0.8$, the departure point would be closer to the additional moment of CSL3U ($\hat{f}_{i-1}$ in Fig. 3.2). The numerical result of CSL3D and CSL3U are symmetrical at $CFL = 0.5$ since the departure point in this case is at equal distance to the additional moments in both cases. This finding confirms the Fourier analysis done in Section 3.3.

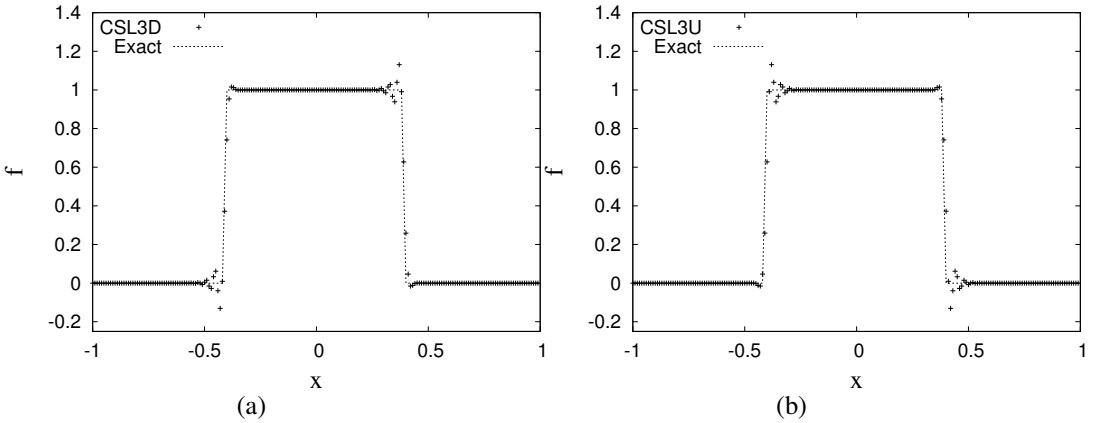


Figure 3.10: Numerical results of square wave propagation at $t = 1$ (500 time steps) with $CFL = 0.5$.

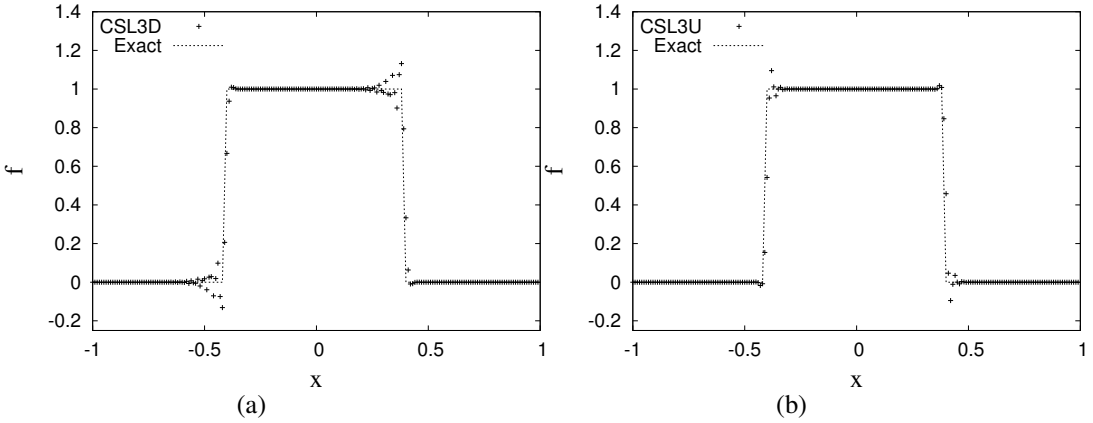
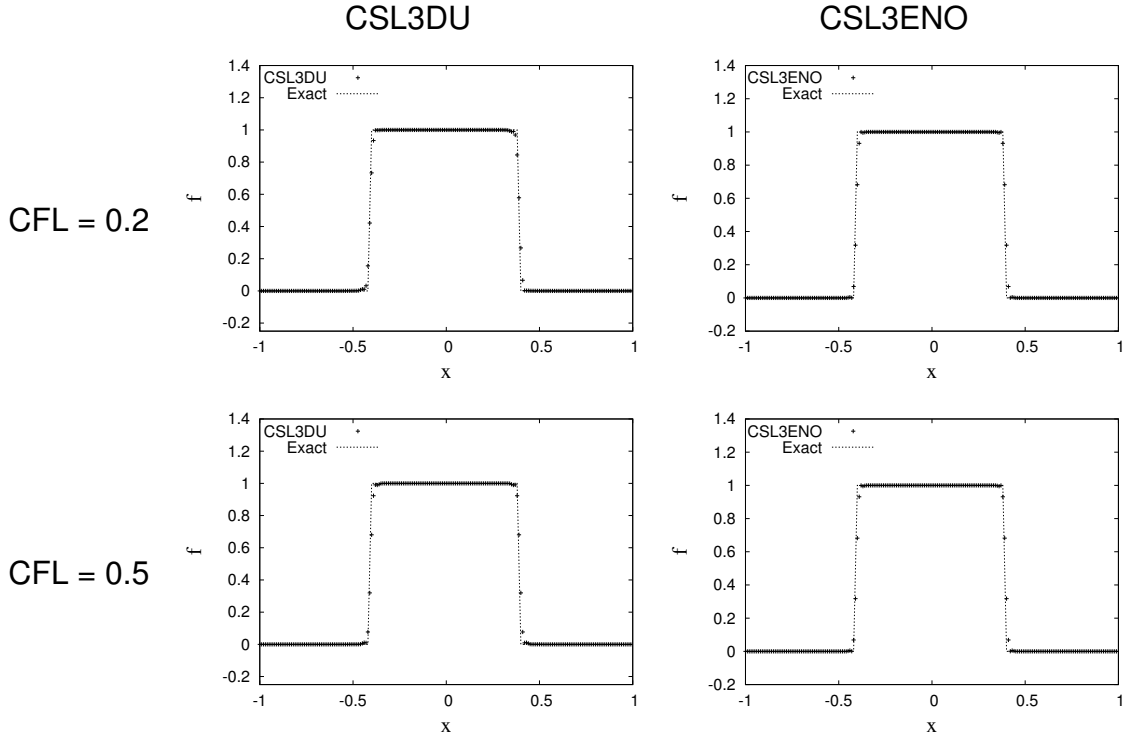


Figure 3.11: Numerical results of square wave propagation at $t = 1$ (500 time steps) with $CFL = 0.8$.



3.6.3 Complex wave

The proposed methodology is tested using the Jiang-Shu complex wave propagation problem [82], which contains a combination of Gaussian, a square wave, a shard triangle wave, and an ellipse. The velocity is set as $u(x) = 1$, grid size $N = 200$, $\Delta t = 0.4\Delta x$, $\Delta x = 2/N$, using periodic boundary conditions with computational domain [-1:1]. The initial conditions are given as follows

$$f(x,0) = \begin{cases} \frac{1}{6}\left(G(x,\beta,z-\delta) + G(x,\beta,z+\delta) + 4G(x,\beta,z)\right) & -0.8 \leqslant x \leqslant -0.6, \\ 1 & -0.4 \leqslant x \leqslant -0.2, \\ 1 - |10(x-0.1)| & 0.0 \leqslant x \leqslant 0.2, \\ \frac{1}{6}\left(F(x,\alpha,a-\delta) + F(x,\alpha,a+\delta) + 4F(x,\alpha,a)\right) & 0.4 \leqslant x \leqslant 0.6, \\ 0 & \text{otherwise} \end{cases} \tag{3.51}$$

where

$$G(x,\beta,z) = e^{-\beta(x-z)^2}, \tag{3.52}$$

$$F(x,\alpha,a) = \sqrt{\max(1-\alpha^2(x-a)^2, 0)}, \tag{3.53}$$

here $a = 0.5$, $z = -0.7$, $\delta = 0.005$, $\alpha = 10$ and $\beta = \log(2)/(36\delta^2)$.

Results at 4000 time steps show that CSL3DU and CSL3ENO could capture all the profiles well compared to CSL2 which, while capturing relatively well the Gaussian wave, has oscillations around discontinuity. CSL3CW managed to suppress all oscillations but overcompensated in this direction and therefore became diffusive. Results at 40,000 time steps show that the profiles captured by CSL3DU and CSL3ENO are still relatively well-preserved, with CSL3DU performing slightly better. At this point CSL2 and CSL3CW has lost any semblance to the original profile.

Table 3.2: Errors in the complex wave propagation at t=16 (after 4,000 time steps) when N=200 is used.

	L_1 error	L_∞ error
CSL2	4.18×10^{-2}	4.21×10^{-1}
CSL3CW	5.87×10^{-2}	4.59×10^{-1}
CSL3D	3.19×10^{-2}	4.59×10^{-1}
CSL3U	3.72×10^{-2}	4.67×10^{-1}
CSL3DU	2.42×10^{-2}	3.65×10^{-1}
CSL3ENO	2.30×10^{-2}	3.86×10^{-1}

Table 3.3: Errors in the complex wave propagation at t=160 (after 40,000 time steps) when N=200 is used.

	L_1 error	L_∞ error
CSL2	9.29×10^{-2}	4.58×10^{-1}
CSL3CW	1.47×10^{-1}	6.79×10^{-1}
CSL3D	6.32×10^{-2}	5.14×10^{-1}
CSL3U	6.61×10^{-2}	5.45×10^{-1}
CSL3DU	4.12×10^{-2}	4.14×10^{-1}
CSL3ENO	4.43×10^{-2}	4.31×10^{-1}

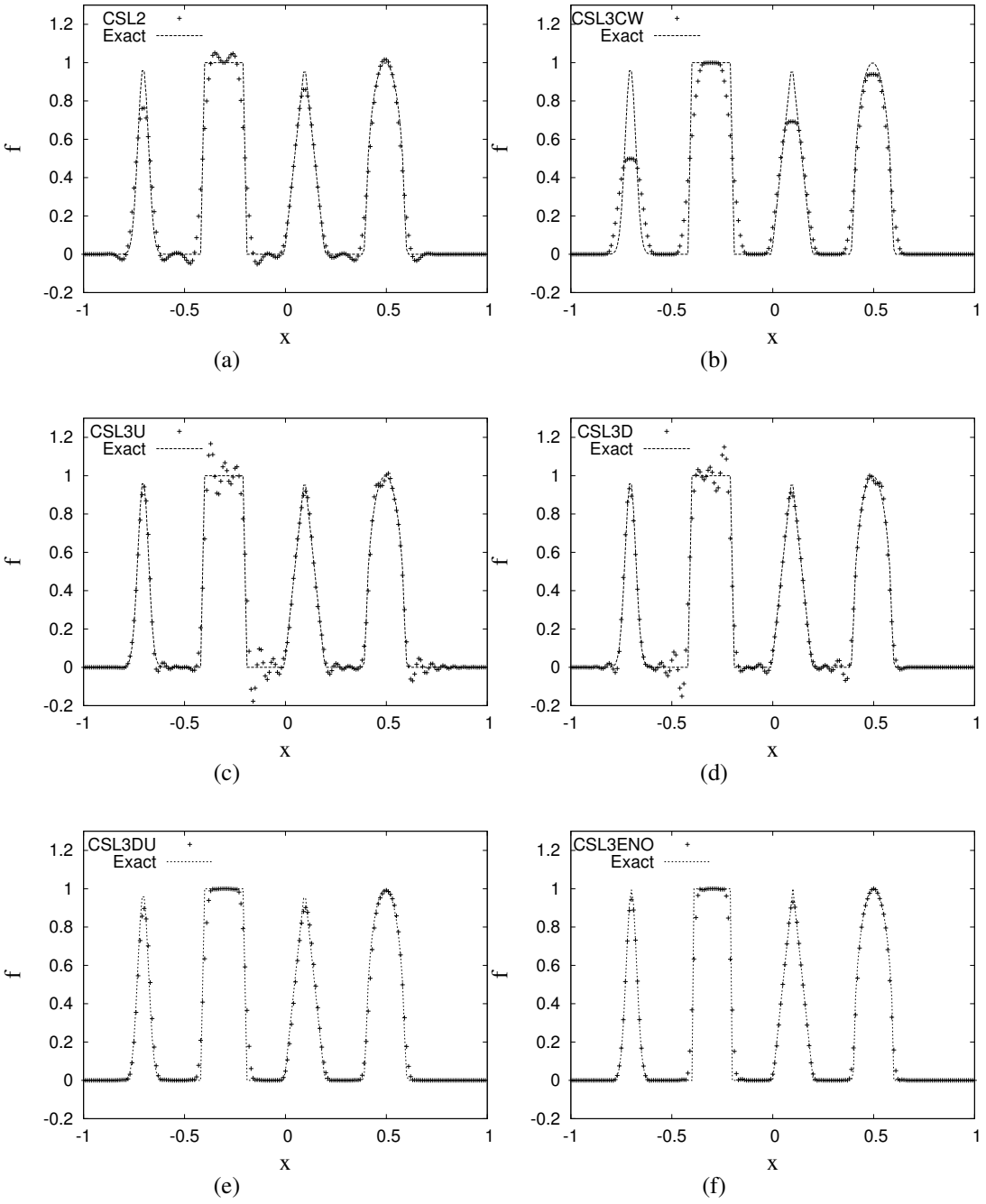


Figure 3.13: Numerical results of complex wave propagation at 4000 time steps

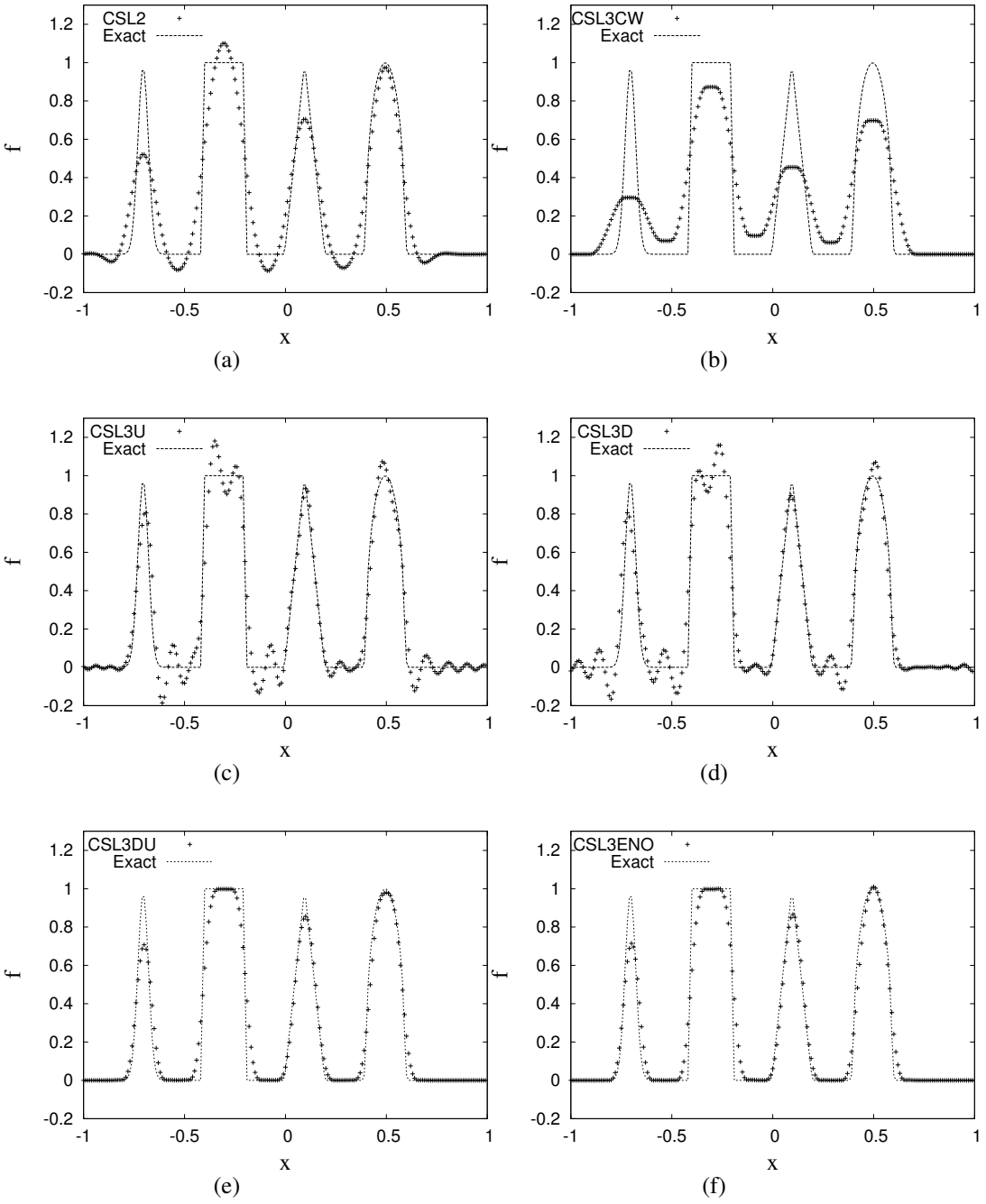


Figure 3.14: Numerical results of complex wave propagation at 40,000 time steps

3.6.4 Extrema of various smoothness

The proposed scheme is verified against a test proposed by Harten et al. [80] called the extrema of various smoothness capture test. The mesh size is $N = 100$, $\Delta x = 2/N$, timestep $\Delta t = 0.4\Delta x$, and velocity $u = 1$. Periodic boundary conditions are used for the computational domain [-1.5:0.5]. The initial conditions are as follows

$$f(x+0.5,0) = \begin{cases} -x\sin(1.5\pi x^2) & \text{for } -1 \leq x < -\frac{1}{3} \\ |\sin(2\pi x)| & \text{for } |\frac{1}{3}| \geq x \\ 2x - 1 - \frac{\sin(3\pi x)}{6} & \text{otherwise .} \end{cases} \tag{3.54}$$

The results of this test are consistent to Subsection 3.6.3. After four cycles, CSL2 managed to capture the profile relatively well while CSL3CW is again very diffusive. CSL3DU and CSL3ENO reproduced the profile with neither the diffusion present in CSL3CW nor the slight oscillations of CSL2.

Table 3.4: Errors in the extrema of various smoothness at t=8 when N=100 is used.

	L_1 error	L_∞ error
CSL2	6.41×10^{-2}	8.18×10^{-1}
CSL3CW	1.14×10^{-1}	9.15×10^{-1}
CSL3D	6.64×10^{-2}	8.09×10^{-1}
CSL3U	6.83×10^{-2}	7.88×10^{-1}
CSL3DU	4.01×10^{-2}	7.49×10^{-1}
CSL3ENO	4.13×10^{-2}	7.24×10^{-1}

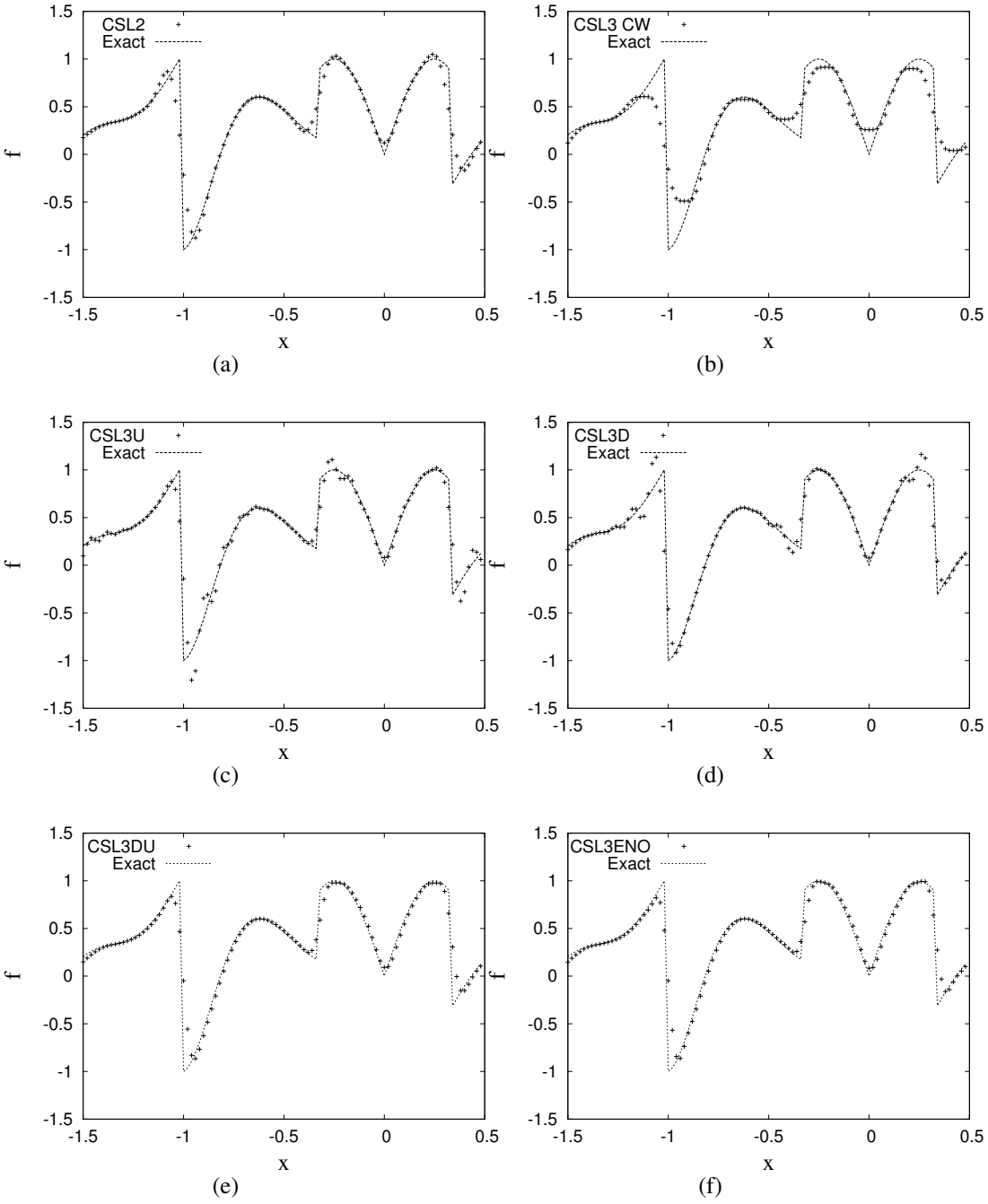


Figure 3.15: Numerical results of extrema of various smoothness test at 1000 time steps (4 cycles).

3.6.5 Non-uniform velocity test

A square wave is transported in the following non-uniform velocity field. The initial conditions are as follows;

$$f(x,0) = \begin{cases} 1 & \text{for } 0.35 \leq x < 0.65 \\ -1 & \text{otherwise ,} \end{cases} \tag{3.55}$$

$$u(x,0) = \frac{1}{1 + 0.4\sin(2\pi x)} \tag{3.56}$$

with mesh size $N = 300$, $\Delta x = 1/N$, $\Delta t = 0.2\Delta x$. Results show that CSL3DU performs well even in a non-uniform velocity field. CSL2 produces oscillations on both sides of the discontinuity and CSL3CW is too diffusive. CSL3D and CSL3U shows oscillation occurring on opposite sides of discontinuities. CSL3DU captured the profile with minimal oscillation and diffusion. CSL3ENO has very minor oscillation at the discontinuity as can be seen in Fig. 3.16(f), but is slightly sharper than CSL3DU.

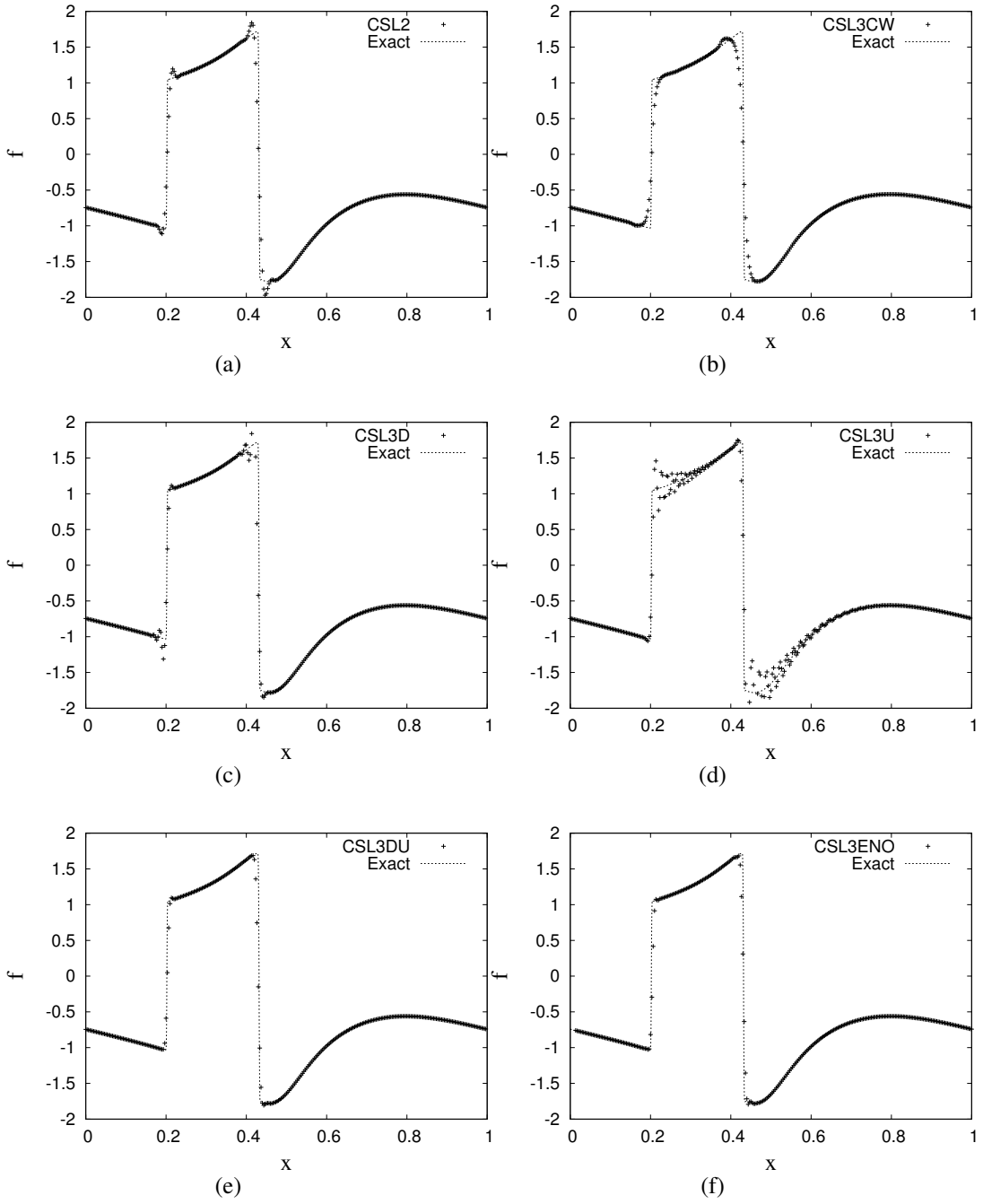


Figure 3.16: Numerical results of the density profile for the non-uniform velocity test at $t = (1.8/dt)$.

3.6.6 Burgers equation

In this test by Qiu and Shu [103], the proposed scheme is evaluated by solving the non-linear inviscid Burgers equation which appears in the studies of gas dynamics;

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0 \tag{3.57}$$

which, in its conservative form is

$$\frac{\partial u}{\partial t} + \frac{\partial (u^2/2)}{\partial x} = 0 \tag{3.58}$$

Eq. (3.58) is solved with the smooth initial condition

$$u(x, 0) = 0.5 + 0.4\cos(2\pi x). \tag{3.59}$$

A crucial phenomenon brought about by Burgers equation is the formation of shocks, which are discontinuities that arise after an amount of time, later propagating in a regular manner. The reference solution is created using the CSL3 method with N = 1000. It can be seen in this test that CSL2, CSL3D, and CSL3U have some oscillations at the shock line, with CSL2 being the worst affected. CSL3CW managed to capture the profile relatively well but has some diffusion. The same can be said about CSL3DU. CSL3ENO captured the discontinuity very well, giving way to very minimal diffusion.

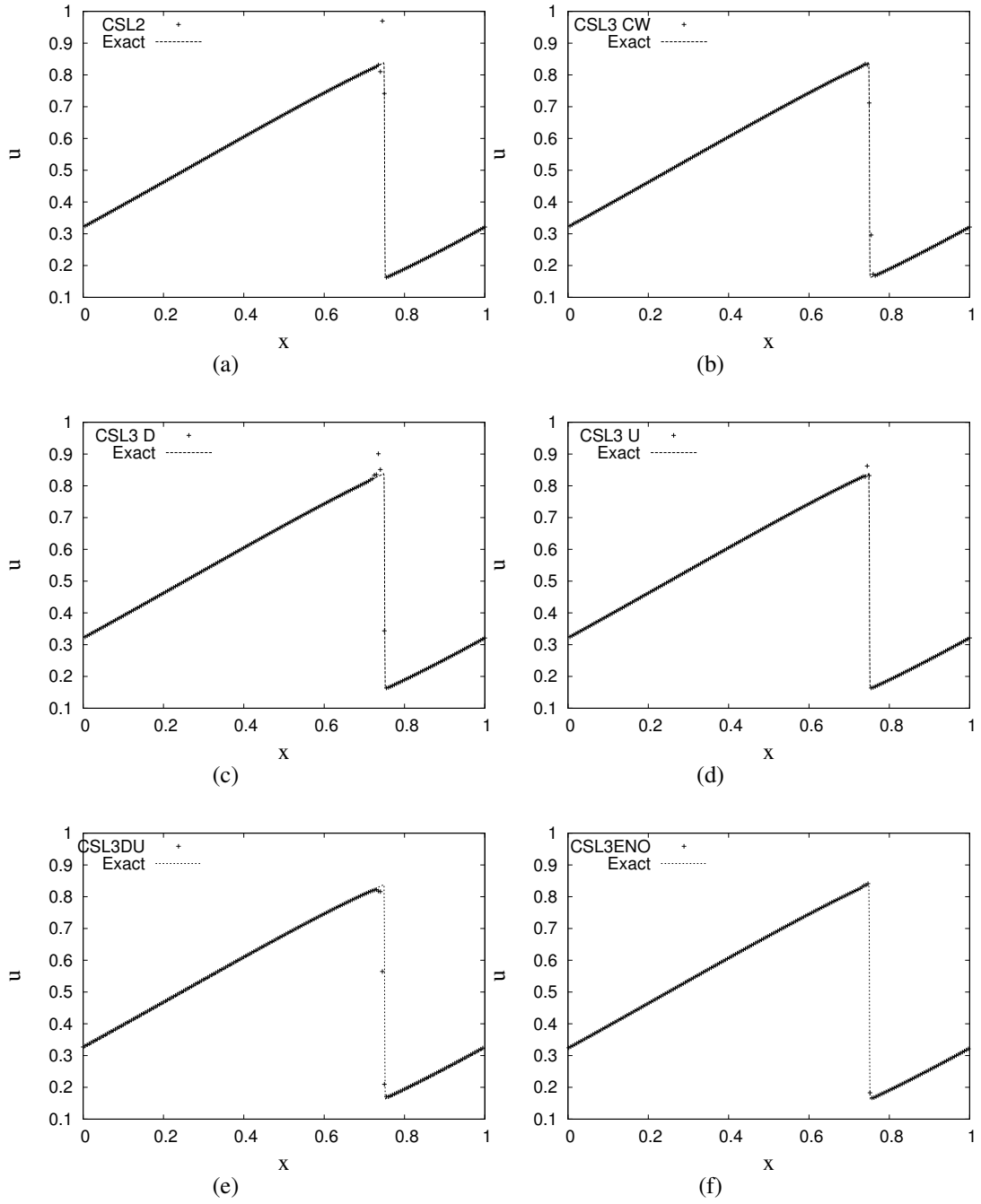


Figure 3.17: Numerical results of Burger's equation at t=1 using N=200

3.6.7 Sod's problem

From this point onwards, the schemes are tested for compressible flow problems using the third-order Runge-Kutta characteristic formulation [91], detailed in Appendix (6.2). Sod's problem [104] is a well-known benchmark for one-dimensional Euler equations problems. It is based on a one-dimensional shock tube with a thin diaphragm placed in the middle to separate a high pressure region and a lower pressure region as in Fig. 3.18. At $t = 0$, the gas is at rest. Flow is generated when the diaphragm is instantaneously removed. The initial conditions are;
if $x \leq 0.5$:

$$\rho(x,0) = 1; \quad u(x,0) = 0; \quad p(x,0) = 1 \tag{3.60}$$

otherwise:

$$\rho(x,0) = 0.125; \quad u(x,0) = 0; \quad p(x,0) = 0.1 \tag{3.61}$$

using Dirichlet boundary conditions.

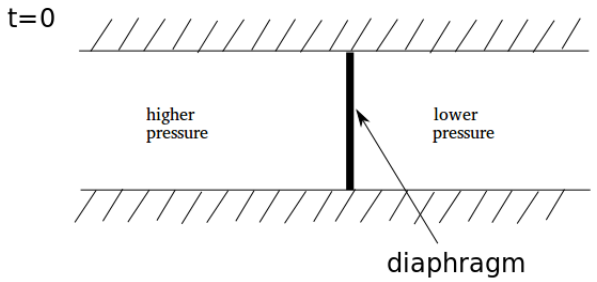


Figure 3.18: Initial condition of shock tube problem

Results from the Sod test indicate that CSL2 is oscillatory around discontinuities. CSL3D and CSL3U exhibit oscillation on the upwind and downwind side of the discontinuity, respectively. CSL3CW is diffusive. CSL3DU and CSL3ENO both managed to capture the contact discontinuity as well as the shock wave with minimal diffusion and oscillation.

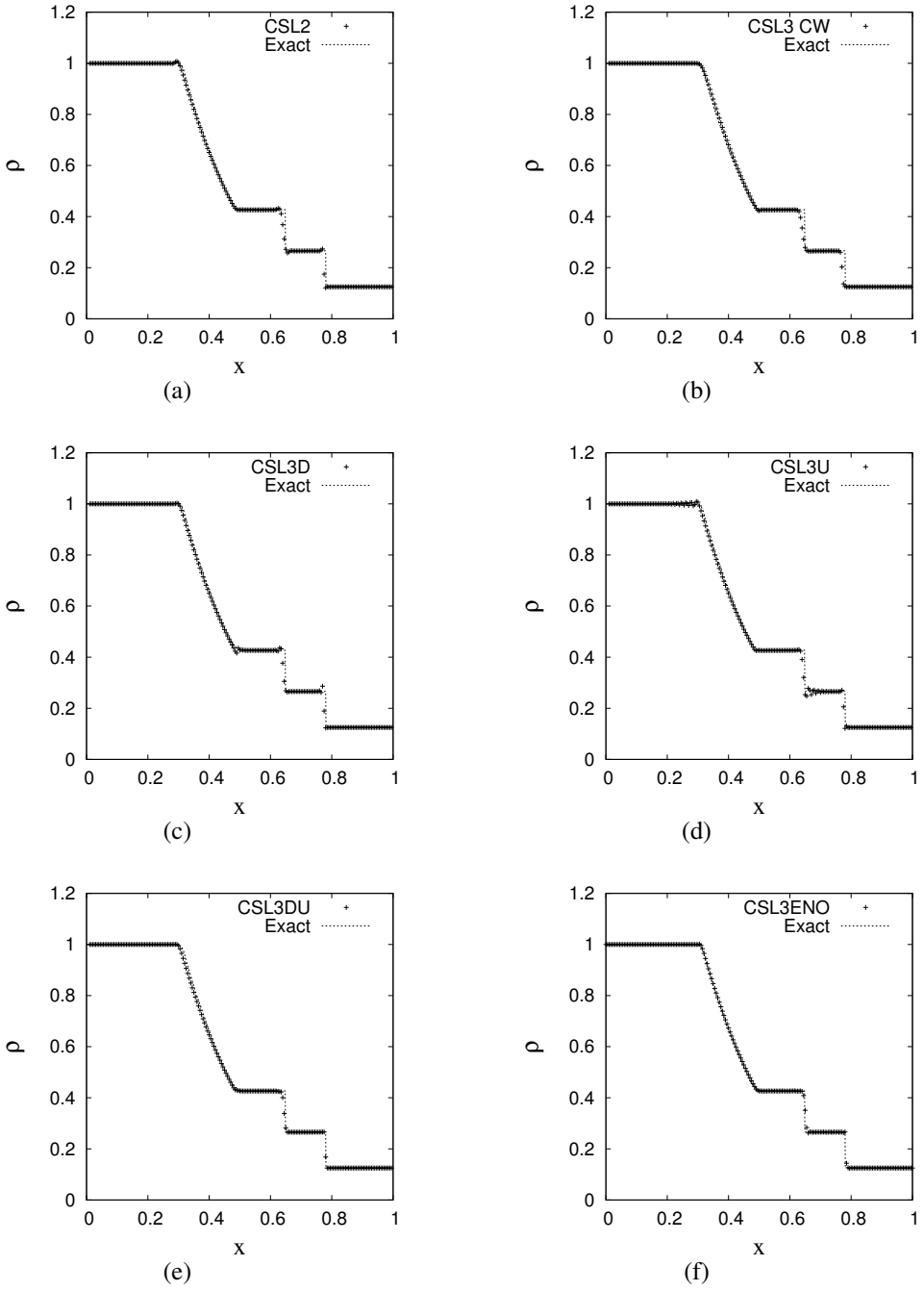


Figure 3.19: Numerical results the density profile for Sod's problem at $t = 0.16$ with $N = 200$ with $CFL = 0.2$.

3.6.8 Lax's problem

Proposed by Lax in 1954 [71], this test is similar to the Sod's test but contains a stronger shock and contact discontinuity. The initial conditions are given as follows;
if $x \leq 0.5$:

$$\rho(x,0) = 0.445; \quad u(x,0) = 0.698; \quad p(x,0) = 3.528 \tag{3.62}$$

otherwise:

$$\rho(x,0) = 0.5; \quad u(x,0) = 0; \quad p(x,0) = 0.571 \tag{3.63}$$

The Lax test has shown that CSL2 continues its oscillatory trend near discontinuities. CSL3D and CSL3U are highly oscillatory at the upwind and downwind side of the discontinuity, respectively. CSL3CW captured the general outline of the solution with no oscillation, but is relatively diffusive compared to CSL3DU and CSL3ENO. Both of the proposed methods managed to capture the profile with minimal diffusion. However, some oscillation could be observed on either side of the discontinuity for CSLDU, and to a lesser extent, along the top of the profile for CSL3ENO.

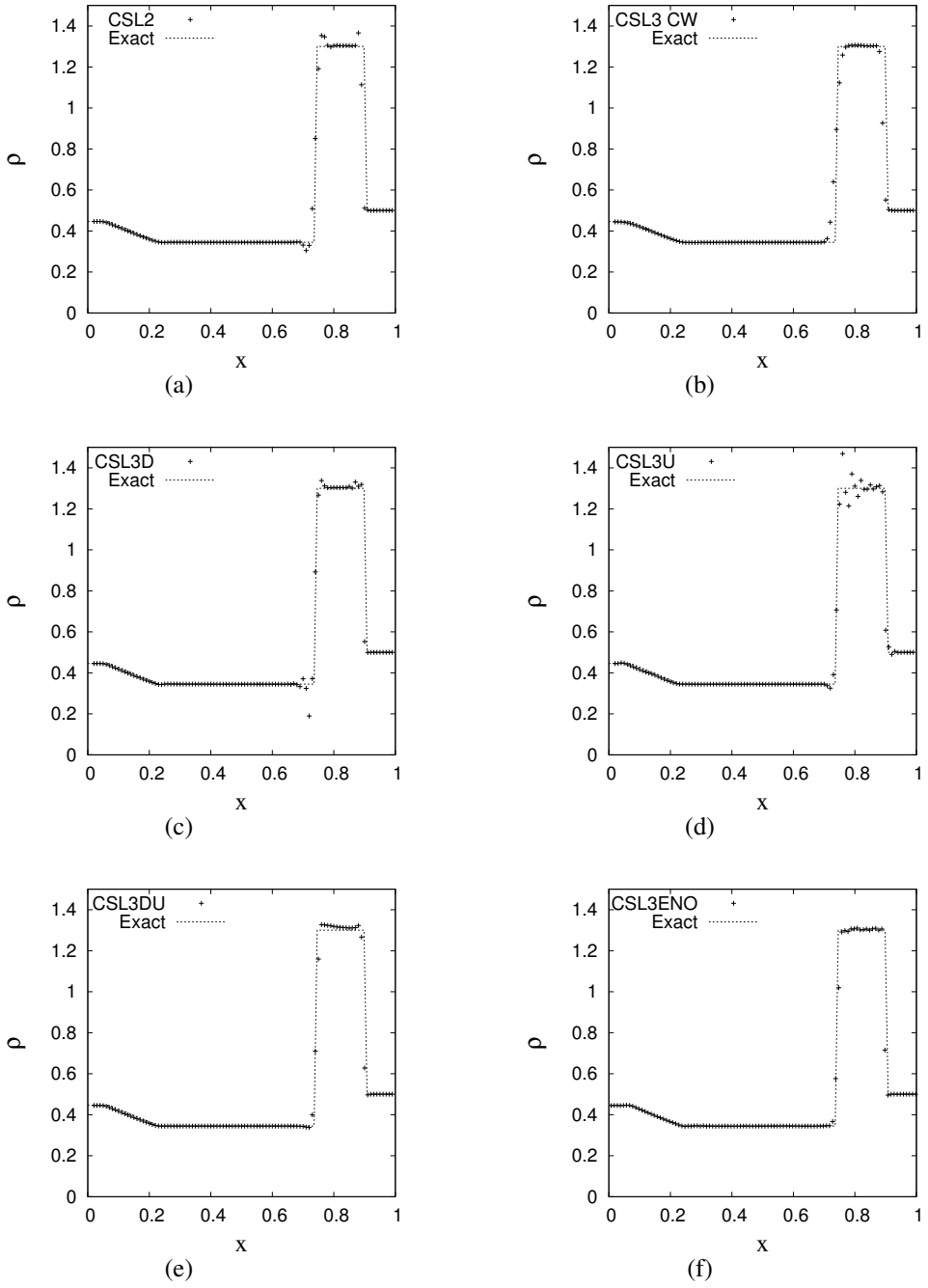


Figure 3.20: Numerical results of density profile for Lax's problem at $t = 0.2$ with $N = 100$ with $CFL = 0.2$.

3.7 Conclusion

A new variant of the CIP-CSL3 solver, called CIP-CSL3U, was proposed as a complement to an existing scheme, CIP-CSL3D. The CSL3D scheme has an extra constraint in the downwind cell, and CSL3U has a constraint on the upwind side. As a result, CSL3D produces numerical oscillation on the side that is upwind to the discontinuity, and vice versa for CSL3U. Due to the symmetrical nature of these two schemes, an ENO-like approach is taken to automatically select the smoother stencil during runtime. This requires a selector that can identify and select the smoother stencil.

Two selectors are proposed in this work, each producing the two schemes CIP-CSL3DU and CIP-CSL3ENO. The first selector (CSL3DU) chooses the smoother stencil as indicated by the proposed smoothness indicator. The second selector (CSL3ENO) intentionally selects the stencil that contains opposing slopes.

Benchmark tests indicate that CSL3DU and CSL3ENO retained a high-order of accuracy of almost 4th order for sine wave tests. Tests with complex profiles show that CSL3DU and CSL3ENO perform similarly, eliminating almost all oscillations while retaining sharpness. Both the proposed schemes outperform the other schemes compared in this work. Additional tests are carried out for compressible flow problems, which show CSL3DU and CSL3ENO outperforming the other schemes, with CSL3ENO retaining a slight edge over CSL3DU. The success of the tests show that the proposed ENO-formulation for CIP-CSL, of selecting between two inherently oscillatory but complementary stencils, has potential for the application of incompressible and compressible flow problems.

Chapter 4

Interface Capturing with Geometrical CLSVOF

In this chapter, the methodology and implementation of a new CLSVOF solver based on the isoAdvector method in interFlow [2] for two-phase incompressible, immiscible flows on the OpenFOAM platform is presented.

Firstly an overview of the algorithm is given. This is then detailed in a step-by-step manner with particular focus on the interface reconstruction algorithm, which includes algorithms for finding the volume of a general polyhedron cut by a plane. The adaptation of the isoAdvector scheme is then detailed, followed by the reinitialisation procedure for the LS field.

Finally the implementation is verified against several validation tests on structured and unstructured meshes of various polyhedra types.

4.1 Implementation of CLSVOF on general meshes

An overview of the algorithm used in this implementation is given as in Fig. 4.1. Firstly the domain is initialised with the VOF (α) and LS (ϕ) fields. The interface is reconstructed inside each cell using the values of the LS field. In order to maintain mass conservation, the position of the interface in the cell is adjusted to match exactly

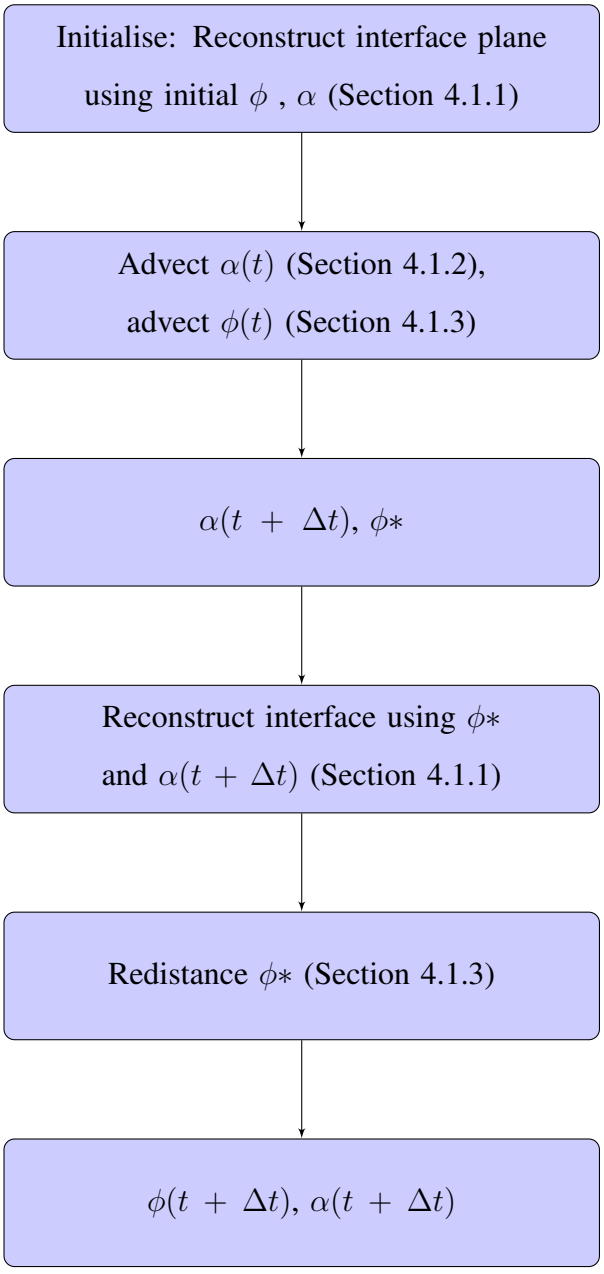


Figure 4.1: CLSVOF interface capturing method overview

the volume in the cell as given by the VOF field. Both fields are advected with the velocity field. The interface is reconstructed and adjusted again, after which the LS field around the interface is redistanced. The following subsections will now go through the procedure.

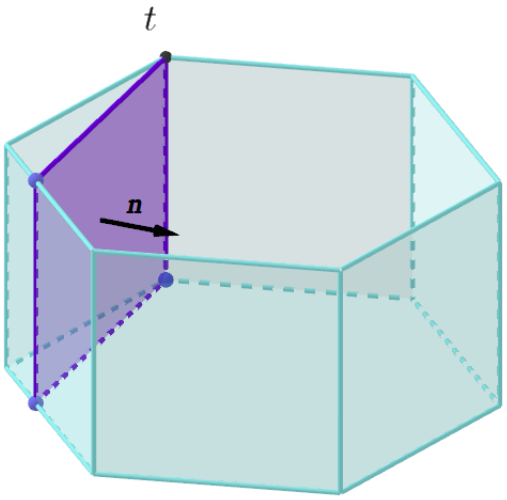


Figure 4.2: A plane signifying the interface in a cell at time t, with normal $\mathbf{n}$

4.1.1 Interface reconstruction algorithm for general polyhedra

A flat 2 D plane signifying the boundary between two fluids in an interface cell is to be constructed. This plane is perpendicular to the normal vector of the liquid interface in each cell (Fig. 4.2). The equation of the plane (and therefore its orientation) is found using the LS field values of the cell in question as well as its neighbouring cells, and the position of the plane is adjusted along the normal such that the volume of Fluid L underneath the plane matches exactly with the volume prescribed by the VOF field. This is the paramount feature of the geometric CLSVOF method that ensures mass-conservation as well as a robust representation of the fluid interface.

A drawback of this procedure is that it is complex to code in a general unstructured 3D mesh, as the reconstructed plane could be oriented in any manner inside the cell, making it difficult to calculate the volume of fluid under the plane. To handle this issue, in this work, a tetrahedralisation process is applied to interface cells, to calculate the volume of a general convex polyhedron intersected by a plane. In this method, all convex polyhedra containing a fluid interface are reduced into tetrahedra, which simplifies the volume calculation.

4.1.1.1 Identifying interface cells

The interface is reconstructed only in cells that contain more than one fluid phase. These cell are identified as follows, as per [105] [45] [46]:

Algorithm 4.1: Identifying interface cell

for all cell i **do**
  if ($\phi_i \; \phi' < 0$) **then**
    if ($0 < \alpha_i < 1$) AND ($\phi_i \; (\phi_i \; \phi') < 0$) **then**
      cell i is an interface cell
    end if
  end if
end for

where ϕ_i and α_i are the field values in the cell where the interface is being reconstructed, and ϕ' is the LS value in the first layer of neighbouring cells.

4.1.1.2 Building an interface plane in each interface cell

The fluid interface plane in each interface cell i is represented by the linear function

$$\phi_i^R = A_i(x - x_i) + B_i(y - y_i) + C_i(z - z_i) + D_i. \tag{4.1}$$

where (A_i, B_i, C_i) is the normal vector and D_i is the Euclidean distance of the plane to the interface cell centre. This is normalised by

$$\mathbf{n}_\phi = \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} = \begin{pmatrix} A_i/\sqrt{A_i^2 + B_i^2 + C_i^2} \\ B_i/\sqrt{A_i^2 + B_i^2 + C_i^2} \\ C_i/\sqrt{A_i^2 + B_i^2 + C_i^2} \end{pmatrix} \tag{4.2}$$

where $\mathbf{n}_\phi$ is the unit normal of the plane. The coefficients A_i, B_i, C_i, D_i need to be found such that it represents the 0-contour plane of the LS field in that cell. In this

work we perform a gradient-based reconstruction method, where $\mathbf{n}_\phi$ is approximated by the gradient of the LS field;

$$\mathbf{n}_\phi \sim \Big(\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y}, \frac{\partial\phi}{\partial z}\Big). \tag{4.3}$$

In 3D unstructured meshes composed of general polyhedra, a Least-Squares approach is the most convenient one to approximate the gradient [70], using the values in the set of neighbour cells N. The following weighted error function is minimised

$$E_i = \sum_N [w'(\phi' - A_i(x' - x) - B_i(y' - y) - C_i(z' - z) - D_i)]^2 \tag{4.4}$$

where w' is a weight in the form of the inverse of the distance between the cell centres of i and the neighbours [106], ϕ' is the value of LS in the neighbour cells, and x', y', z' are the cell centre coordinates of the neighbour cell. The minimisation of E results in a solution for a system of algebraic equations for each cell which is solved for A_i, B_i, C_i, D_i, producing a second-order accurate gradient regardless of the arrangement of the neighbour points.

4.1.1.3 Decomposing interface cells into constituent tetrahedra

Outline

If the cell is determined to contain a fluid interface, it is decomposed into tetrahedra unless it is already one. This is to provide a convenient data structure to be used later in the algorithm. Fig. 4.3 presents an overview of the resultant tetrahedra for various types of cell. The 'Primary decomposition' column shows the exploded view of the decomposed cell. For clarity, pyramids are used to represent four individual tetrahedron in the cube and prism decompositions. The decomposition is described using more detail in the following sections.

Data structure

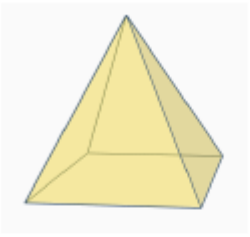
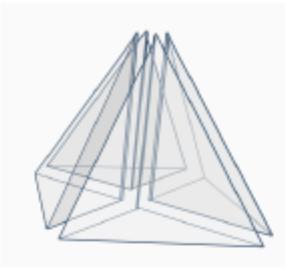
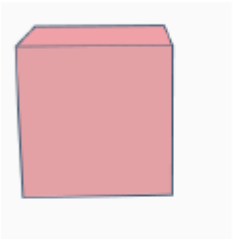
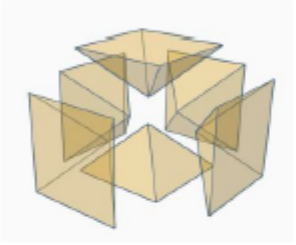
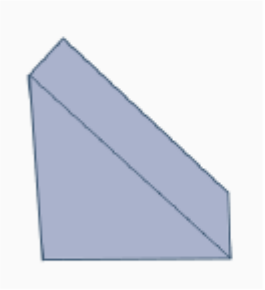
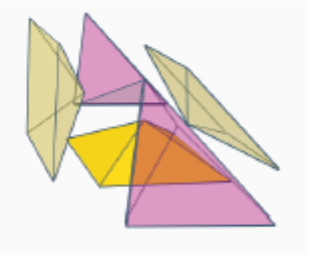
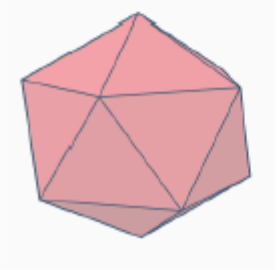
Cell Shape	Primary decomposition	Final tetrahedron count
Pyramid 		4 tetrahedra
Cube 		6 pyramids -> 24 tetrahedra
Prism 		2 tetrahedra, 3 pyramids -> 14 tetrahedra
Icosahedron 	 x 20	20 tetrahedra (one per face)

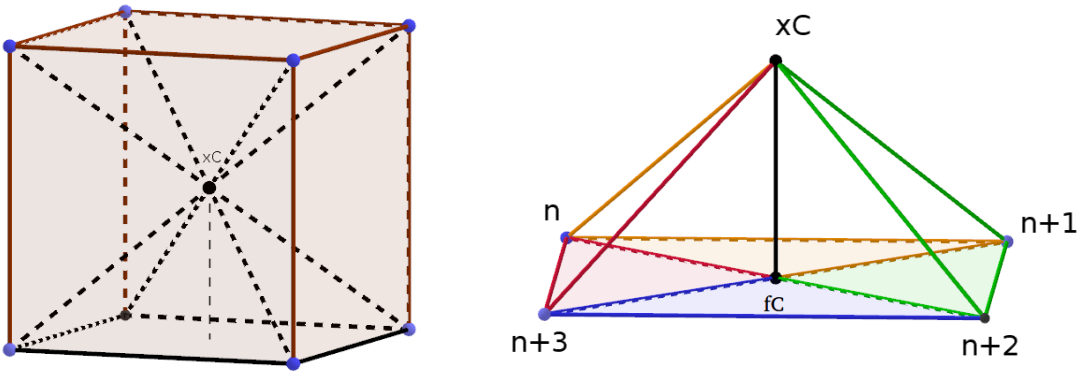
Figure 4.3: Various cell shapes and their decomposition

For each cell containing an interface, its constituent tetrahedra are stored in 'cellDecomposedTetra'. Each of the tetrahedron is stored as a list of its vertices in 'tmpTetra', so 'cellDecomposedTetra' is a collection of this list.

Decomposition

The decomposition takes place only if the cell is not already a tetrahedron. However, tetrahedron cells are also taken into account in the algorithm in order to structure our data consistently. In the situation where the cell is a tetrahedron, its original vertices are stored in this 'tmpTetra', and therefore into 'cellDecomposedTetra'.

However, if the cell is not a tetrahedron, it is decomposed into tetrahedra based on its cell faces. As an example, consider the case of a cubic cell. Fig. 4.4(a) shows a cubic cell decomposed into six pyramids based on its six faces. Each of these pyramids are further decomposed into four tetrahedra with two common vertices amongst them: the cell centroid of the main cell (point **xC** in Fig. 4.4(b)) and the face centroid of the face around which they were decomposed (**fC**). As in Fig. 4.4(b), the four tetrahedra obtained from the pyramid are $[n, n+1, xC, fC]$, $[n+1, n+2, xC, fC]$, $[n+2, n+3, xC, fC]$, and $[n+3, n, xC, fC]$, which are described in terms of their vertices. These sets of vertices are kept in 'cellDecomposedTetra'. A similar process can be performed on any other convex shapes, described in Algorithm 4.2.



(a) (b)

Figure 4.4: (a) A hexahedral cell shown decomposed into 6 pyramids about each face and (b) a magnified view of the first pyramid taken from the bottom face decomposition, further decomposed into 4 tetrahedra.

Algorithm 4.2: Implementation procedure for decomposing a general polyhedron cell.

for all interface cells **do**
  if cell vertex == 4 **then**
    One tetrahedron in 'tmpTetra' is stored in 'cellDecomposedTetra'.
  else
    Identify cell centroid **xC**, store in $tmpTetra[0]$.
    for all cell faces of cell **do**
      if face vertices == 3 **then**
        1) Populate list $tmpTetra[1]$, $tmpTetra[2]$, $tmpTetra[3]$ (Fig. 4.4(a)) with face vertices.
        2) Append 'tmpTetra' into cellDecomposedTetra.
      else
        1) Insert face centroid **fC** into $tmpTetra[1]$.
        for all face vertices **do**
          1) Insert vertex 1 and vertex 2 into $tmpTetra[2]$, $tmpTetra[3]$ (beginning with n and $n + 1$ as in Figure 4.4).
          2) Append tmpTetra into cellDecomposedTetra.
        end for
      end if
    end for
  end if
end for

4.1.1.4 Adjusting the interface to conserve mass

To ensure the conservation of mass, the fluid volume fraction in cell i as cut by the LS plane is matched to the volume fraction given by the VOF field, α_i. This is done by shifting the interface along the normal $\mathbf{n}_\phi$ until the cell volume cut by the plane equals the volume as given by the VOF field.

An iterative algorithm is used to approximate the value of D, that is the distance of the interface to the cell centre that gives a matching cut volume to the provided VOF volume fraction. The algorithm is detailed as in Algorithm 4.3. Firstly, all interface cells would have a plane that signifies the fluid interface. The volume of the fluid under this plane is calculated with the initial value of D, and the initial volume fraction as given by the plane is $\alpha_{iter=0}$. The initial error E_0 is calculated;

$$E_0 = |\frac{\alpha_{iter=0} - \alpha_i}{\alpha_i}|. \tag{4.5}$$

If E_0 is greater than 1×10^{-4}, the algorithm proceeds into a iteration loop where the value of D is adjusted. With each adjustment of D, the plane is displaced along its normal, which changes the volume under the plane in that cell and hence the volume fraction. This volume fraction is denoted α_{iter}. This work follows Maric [107] and Ann and Shashkov [70] in that firstly the secant method is employed for the iterative procedure. It is then switched to the bisection method if the secant method failed to converge. The convergence criterion is $E_V < 1 \times 10^{-4}$, where

$$E_V = |\frac{\alpha_{iter} - \alpha_i}{\alpha_i}|, \tag{4.6}$$

and α_{iter} is the volume fraction obtained based on each adjustment of D, and α_i is the volume fraction of the cell as given by the VOF field. Recall that the definition of volume fraction α is the volume of fluid L in the cell divided by the volume of the mesh cell, so $\alpha_{iter=0}$ and α_{iter} are found by finding the volume under the plane in that cell, divided by the total cell volume.

Algorithm 4.3: Volume matching iterator

for all interface cells **do**

1) Calculate the volume under the plane with current value of D.

2) Calculate the initial error $E_V = E_0$

if $E_V > (1 \times 10^{-4})$ **then**

Do secant method, update value of D.

Calculate E_V from updated α_{iter}.

if Secant method iteration > 20 **and** $E_V > (1 \times 10^{-4})$ **then**

Do bisection method, update value of D.

Calculate E_V from updated α_{iter}.

end if

end if

end for

Calculating the volume under the plane

To calculate α_{iter}, the volume of the fluid under the plane has to be calculated at each iteration as D changes. Calculating this volume is a non-trivial task due to the number of possible configurations available for a plane intersecting a 3D polygon.

Six possible configurations have been identified. This is done by calculating the signed distance of the tetrahedron vertices to the plane. The configuration is classified by the combination of the number of 'submerged' vertices, 'non-submerged' vertices, and vertices that are on the interface plane.

The vertices under the plane is stored as 'submerged points' denoted with index neg for $negative$. These indicate the portion of the cell occupied by fluid L. Inversely the vertices above the plane is denoted with index pos for $positive$. If the tetrahedron vertex is on the plane, it is denoted as $zero$. The six configurations for plane-tetrahedra intersect are classified as follows, using a combination of the number of vertices above, below, and on the plane.

- Case 1: $neg = 1$, $pos = 3$, $zero = 0$.
- Case 2: $neg = 2$, $pos = 2$, $zero = 0$.
- Case 3: $neg = 1$, $pos = 2$, $zero = 1$.
- Case 4: $neg = 3$, $pos = 1$, $zero = 0$.
- Case 5: $neg = 2$, $pos = 1$, $zero = 1$.
- Case 6: $neg = 1$, $pos = 1$, $zero = 2$.

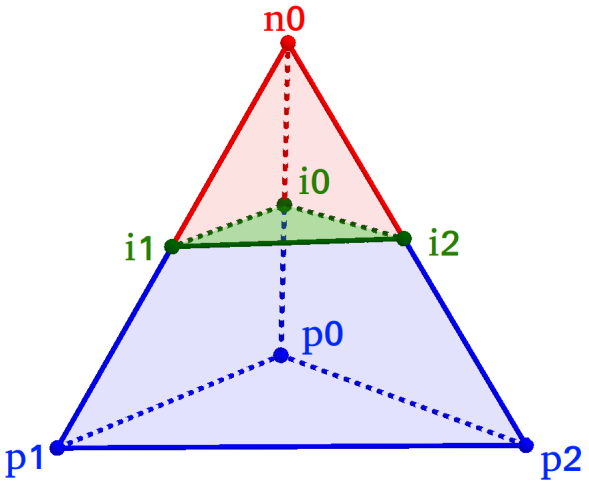


Figure 4.5: Cutting sequence of a Case 1 ($neg = 1$, $pos = 3$, $zero = 0$) tetrahedron where the total submerged volume is denoted in red, the interface in green, and the non-submerged volume in blue.

In **Case 1**, the plane cuts the tetrahedron in a straightforward manner, resulting in one tetrahedron and one discarded polygon, depicted in Fig. 4.5. In the diagrams, the 'negative' points are denoted using $n0, n1, ...$, 'positive' points using $p0, p1, ...$, 'zero' points using $z0, z1, ...$, and interface points using $i0, i1, ..$ to identify the vertices. The submerged tetrahedron is defined as a list of points $[n_0, i_0, i_1, i_2]$, stored in the list 'submergedTetra'. In this particular situation, the volume of the submerged tetrahedron

is simply calculated as

$$\text{Volume of tetrahedron } = \frac{1}{6}[(i_1 - i_0) \times (i_2 - i_0) \cdot (n_0 - i_0)]. \tag{4.7}$$

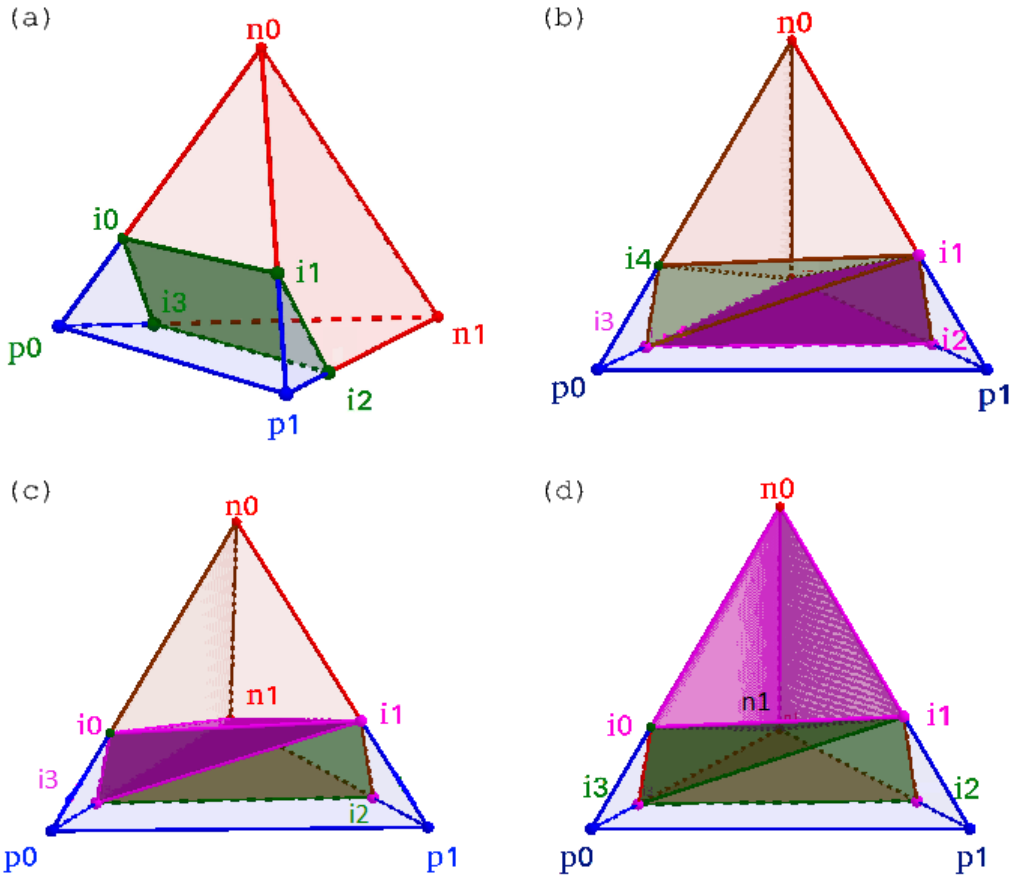


Figure 4.6: Cutting sequence of a Case 2 ($neg = 2$, $pos = 2$, $zero = 0$) tetrahedron where the total submerged area is denoted in red in (a). Its constituent sections are denoted in magenta as in (b), (c), and (d).

For **Case 2** as in Fig. 4.6, the cut results in the submerged volume $[n_0, n_1, i_0, i_1, i_2, i_3]$. This is decomposed further, resulting in three new tetrahedra: $[n_1, i_1, i_2, i_3]$, $[n_1, i_0, i_1, i_3]$, and $[n_0, n_1, i_0, i_1]$. The volumes of these tetrahedra (stored in 'submergedTetra') are simply found similar to Eq. 4.7.

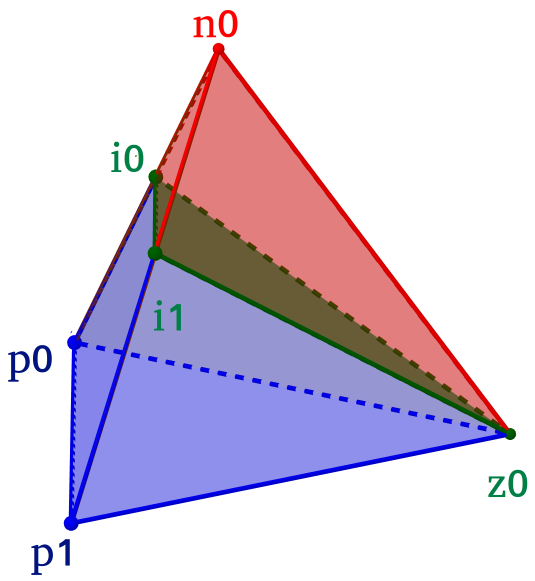


Figure 4.7: Cutting sequence of a Case 3 ($neg = 1$**,** $pos = 2$**,** $zero = 1$) **tetrahedron where the total submerged area is denoted in red.**

Case 3 has a plane cutting directly on a tetrahedron vertex. This directly results in the submerged tetrahedron $[n_0, i_0, i_1, z_0]$, as in Fig. 4.7.

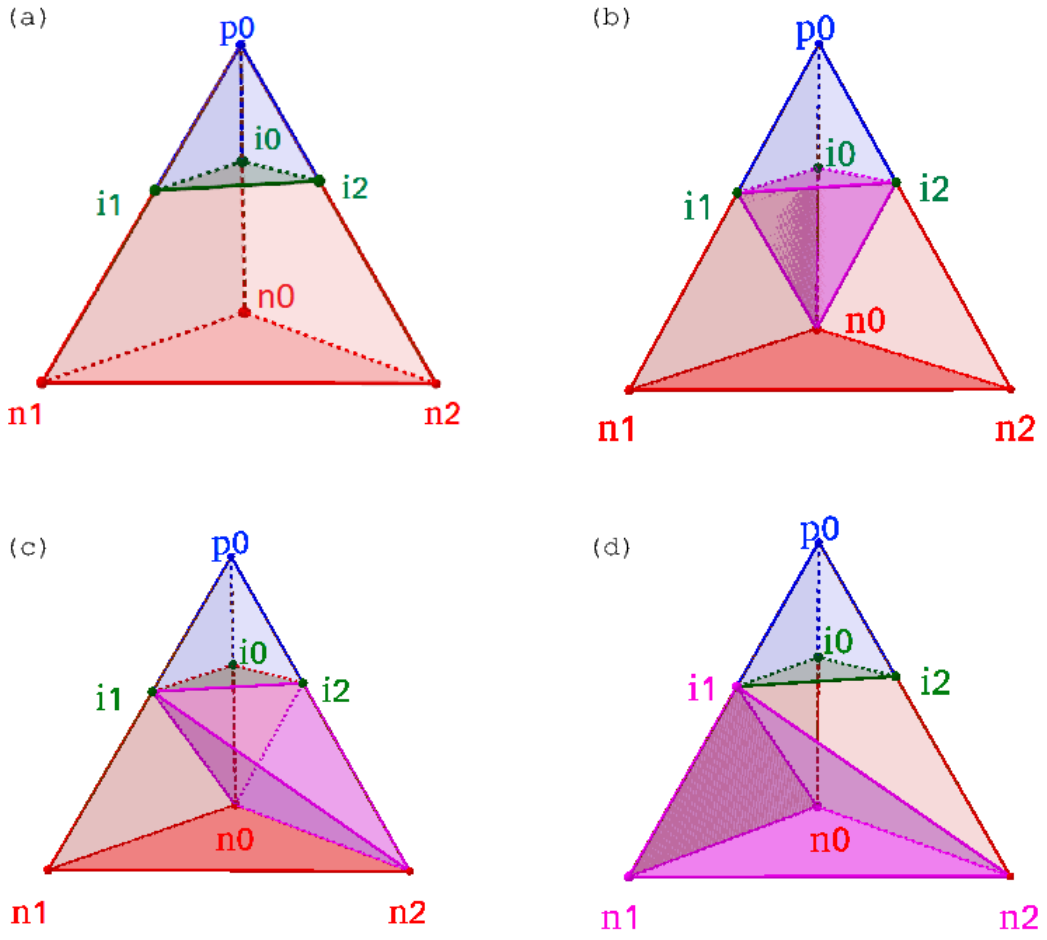


Figure 4.8: Cutting sequence of a Case 4 ($neg = 3$, $pos = 1$, $zero = 0$) tetrahedron where the total submerged area is denoted in red in (a). Its constituent sections are denoted in magenta as in (b), (c), and (d).

Case 4 (Fig. 4.8) results in the submerged section $[i0, i1, i2, n0, n1, n2]$. This is decomposed into three smaller tetrahedra to store in 'submergedTetra': $[n_0, i_0, i_1, i_2]$, $[n_0, n_2, i_1, i_2]$, and $[n_0, n_1, n_2, i_1]$.

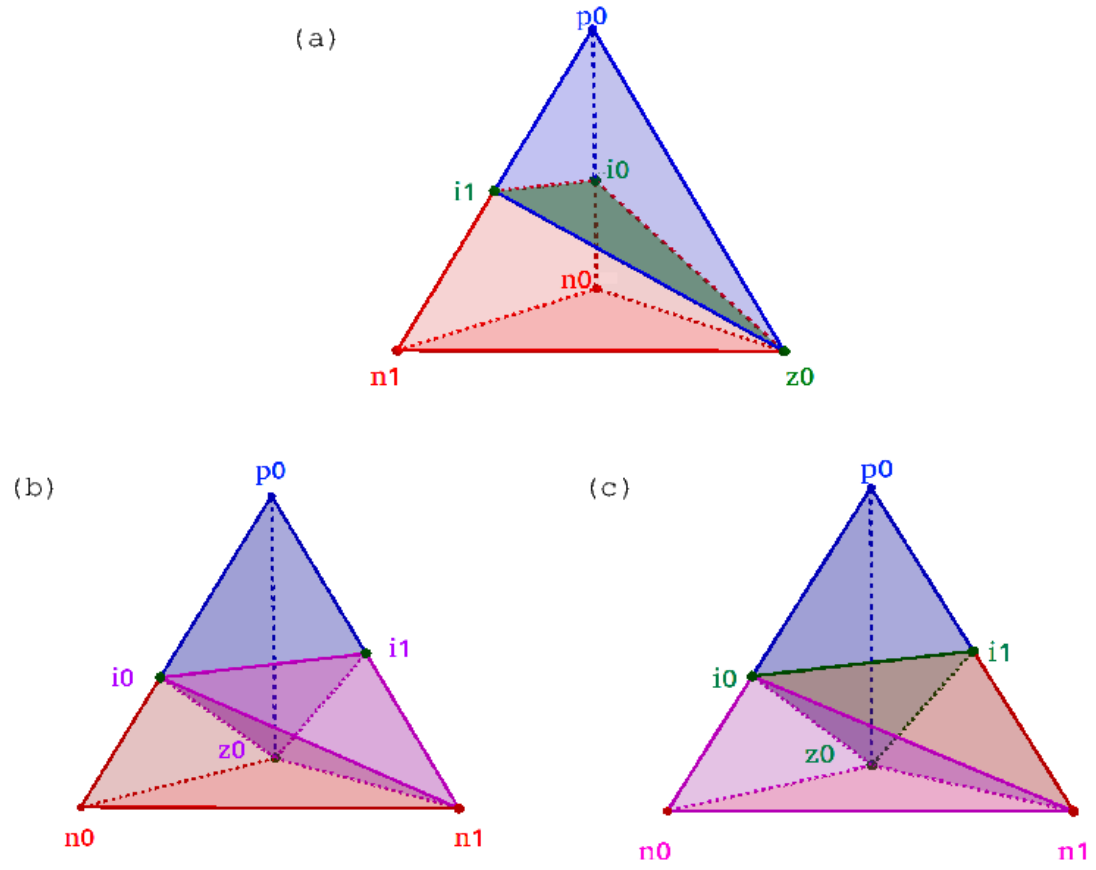


Figure 4.9: Cutting sequence of a Case 5 ($neg = 2$, $pos = 1$, $zero = 1$) **tetrahedron where the total submerged area is denoted in red in (a). Its constituent sections are denoted in magenta as in (b) and (c).**

Case 5 (Fig. 4.9) results in the submerged section $[i0, i1, z0, n0, n1]$. The submerged section is broken down into two tetrahedra: $[z_0, n_1, i_0, i_1]$ and $[z_0, n_0, n_1, i_0]$.

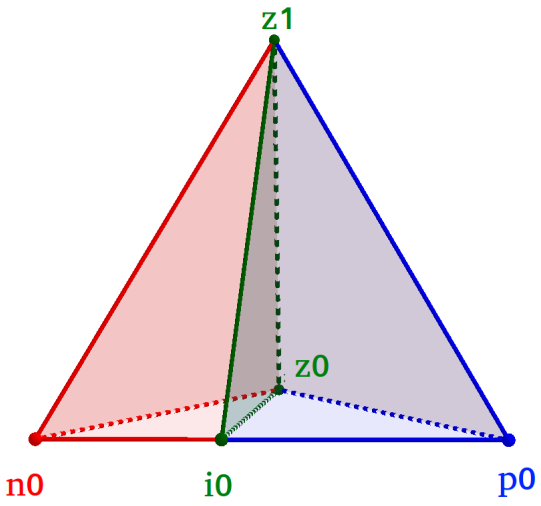


Figure 4.10: Cutting sequence of a Case 6 ($neg = 1$, $pos = 1$, $zero = 2$) tetrahedron where the total submerged area is denoted in red.

Finally, **Case 6** (Fig. 4.10) is relatively straightforward as it yields two tetrahedra, with the submerged one being $[z_0, z_1, i_0, n_0]$.

Example decomposition procedure

As an example for the entire decomposition and cutting procedure, a pyramid cell as in Fig. 4.11 (a) is presented, defined in this section with vertices $[A, B, C, D, E]$. It is decomposed into four tetrahedra using Algorithm 4.2 around two common points, fC and E. These tetrahedra are 'cellDecomposedTetra'. The cell contains an interface which is represented by the plane, with the plane-cell intersects being points $[F, G, H]$ (Fig. 4.11 (b)).

For this example, consider one of the tetrahedra in 'cellDecomposedTetra', $[fC, B, C, E]$ as in Fig. 4.11 (c). The plane-tetrahedra intersects for it are points $[F, G, I]$ (Fig. 4.11 (d)). Depending on the direction of the normal of the plane, the submerged section of this particular tetrahedron is either $[F, G, I, E, B, fC]$ or $[F, G, I, C]$. If the former is the submerged section, it is further decomposed via Case 4 (Fig. 4.8). If the latter

is the submerged section, it is decomposed via Case 1 (Fig. 4.5). This second decomposition yields further tetrahedra 'submergedTetra', and this procedure is repeated for the one remaining tetrahedron in 'cellDecomposedTetra' that is intersected by the plane. To find the volume of the submerged section in this pyramid cell, the volumes of 'submergedTetra' is summed for each 'cellDecomposedTetra'. The tetrahedra in 'cellDecomposedTetra' that are not intersected by the plane i.e. $[A, B, E, fC]$ and $[A, D, E, fC]$ are included in the volume calculation if their vertices are all determined to be 'submerged'.

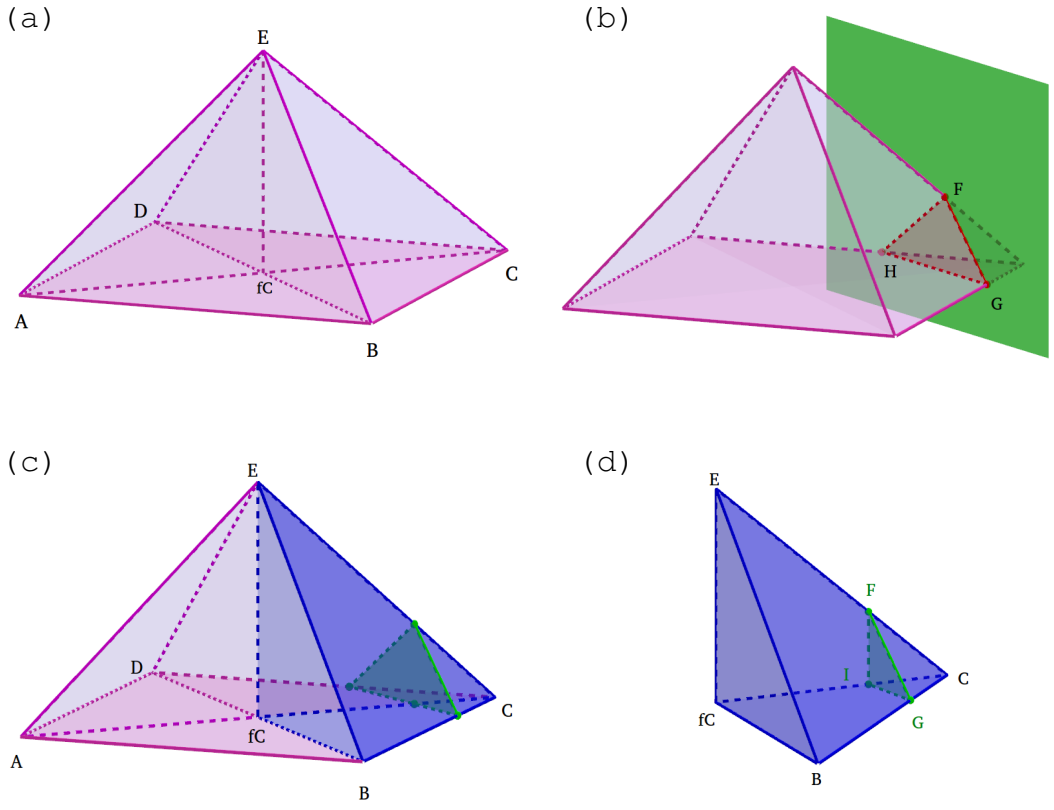


Figure 4.11: A tetrahedron cell is decomposed as in (a) and contains an interface as in (b). The interface cuts across two decomposed tetrahedra, $[fC, B, C, E]$ and $[fC, C, D, E]$ in (c), and the interface intersects the former tetrahedron as in (d), producing three intersect points.

4.1.2 Computing the advected liquid volume fraction from the reconstructed interface

As the orientation and position of the interface that yields a cut volume equal to the VOF fraction is known, the total volume of fluid L transported across face j in the time interval $[t, t + \Delta t]$ can now be calculated to update $\alpha_i(t)$ to $\alpha_i(t + \Delta t)$.

Recall from Section 2.2.2.2 the discretised integral form of the continuity equation:

$$\frac{d}{dt}\int_{C_i} H(\mathbf{x}, t)\, dV = -\sum_{j\in B_i} s_{i,j} \int_{\mathcal{F}_j} H(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S} \tag{4.8}$$

where $H(\mathbf{x}, t)$ is the indicator field, C_i is the discretised cell, B_i is the list of labels belonging to the faces of C_i, $\mathcal{F}_j$ is the surface of face j, and $s_{i,j}$ is either 1 of -1 such that it points out of cell i for face j. The volume fraction α_i is defined at the cell centre, and the volume of fluid L in the cell is then

$$\alpha_i V_i = \int_{C_i} H(\mathbf{x}, t)\, dV, \tag{4.9}$$

with V_i representing the cell volume. Therefore, Eq. (4.8) can be expressed as in Chapter 2, Eq. 2.27;

$$\alpha_i(t + \Delta t) = \alpha_i(t) - \frac{1}{V_i}\sum_{j\in B_i} s_{ij} \int_t^{t+\Delta t} \int_{F_i} H(x, \tau)\, \mathbf{u}(x, \tau) \cdot d\mathbf{S}\, d\tau,$$

to update the value of α_i. Here the time integral on the RHS represents the *total volume of fluid L that is transported across face j between t and $t + \Delta t$*. Since we are applying the isoAdvector method [2], this is the fundamental quantity required to advance α_i in time and it is referred to as $\Delta V_j(t, t + \Delta t)$. The previous equation is now expressed as

$$\alpha_i(t + \Delta t) = \alpha_i(t) - \frac{1}{V_i}\sum_{j\in B_i} s_{ij}\Delta V_j(t, t + \Delta t). \tag{4.10}$$

We recall the definition of $\mathbf{u}(t)$, which is defined at the cell centre

$$\mathbf{u}(t) = \frac{1}{V_i}\int_{C_i} \mathbf{u}(\mathbf{x}, t)\, dV, \tag{4.11}$$

and the volumetric flux across the cell faces,

$$\mathbf{F}_j(t) = \int_{\mathcal{F}_j} \mathbf{u}(\mathbf{x}, t) \cdot d\mathbf{S}. \tag{4.12}$$

This work operates under the same assumptions as the isoAdvector method; the local radius of curvature is larger than the cell size so the interface can be approximated by a flat plane, and the velocity field is constant in time between $[t, t + \Delta t]$, at the intermediate time step τ. Therefore we can write $\mathbf{u}(\mathbf{x}, \tau) \approx \mathbf{u}(\mathbf{x}, t)$. Another assumption following [2] is that the velocity $\mathbf{u}$ on face $\mathcal{F}_j$ can be approximated in terms of $\mathbf{F}_j(t)$,

$$\mathbf{u} \cdot d\mathbf{S} \approx \frac{\mathbf{F}_j(t)}{|\mathbf{S}_j|} dS \tag{4.13}$$

where $dS \equiv d|\mathbf{S}|$ and the face normal $\mathbf{S}_j = \int_{F_j} d\mathbf{S}$. Now $\Delta V_j(t, t + \Delta t)$ can be expressed as

$$\Delta V_j(t, t + \Delta t) \approx \frac{\mathbf{F}_j(t)}{|\mathbf{S}_j|} \int_t^{t+\Delta t} \int_{\mathcal{F}_j} H(\mathbf{x}, \tau)\, d\mathbf{S}\, d\tau. \tag{4.14}$$

The surface integral in Eq. (4.14) is the instantaneous area of face j that is submerged by Fluid L, denoted as $A_j(\tau)$,

$$A_j(\tau) \equiv \int_{\mathcal{F}_j} H(\mathbf{x}, \tau)\, dS. \tag{4.15}$$

With this in mind, the isoAdvector advection method estimates directly the time evolution of the *submerged area of a face* within a time step , and this area $A_j(\tau)$ is integrated in time.

Now the total volume of fluid L transported across face j can be rewritten as

$$\Delta V_j(t, t + \Delta t) \approx \frac{\mathbf{F}_j(t)}{|\mathbf{S}_j|} \int_t^{t+\Delta t} A_j(\tau)\, d\tau. \tag{4.16}$$

$A_j(\tau)$ is a function of the orientations of the face/interface intersect, and this is a function of the motion of the interface and the shape of the cell face.

In order to calculate $A_j(\tau)$ for face j, we start with the interface in the cell upwind of face j at a given time; this cell would be the one to 'donate' the fluid to face j. The

motion of the interface during $[t, t+\Delta t]$ can be estimated by using the velocity data of the neighbouring cells. As the interface is known, the face-interface intersection line is also known for face j in this time period. The time integral in Eq. 2.35, $\int_t^{t+\Delta t} A_j(\tau)\, d\tau$ can then be calculated analytically to obtain the total volume of fluid L transported across face j during $[t, t+\Delta t]$, which is $\Delta V_j(t, t+\Delta t)$.

The algorithm summarising the advection step is given ahead of the detailed description, which will be given in the Sub-subsections to follow.

Algorithm 4.4: Advancing $\alpha_i(t)$ to $\alpha_i(t+\Delta t)$ using the isoAdvector method

for all cells **do**
  for all cell faces j **do**
    Initialise ΔV_j as $\alpha_{upwind(j)}\mathbf{F}_j\Delta t$, where $\alpha_{upwind(j)}$ is the upwind cell volume fraction.
  end for
  for all interface cell i **do**
    1) Find the intersection of the interface plane and the cell face.
    2) Estimate the interface motion in the interval $[t, t+\Delta t]$ (Section 4.1.2.1).
    for all downwind face j of cell i **do**
      1) Calculate the motion of the cell face/liquid interface intersect between $[t, t+\Delta t]$ (Section 4.1.2.2).
      2) Use the motion to calculate $\Delta V_j(t, t+\Delta t)$.
    end for
  end for
  Calculate $\alpha_i(t+\Delta t)$ for all cells using the ΔV_j values of its faces.
  if $\alpha_i(t+\Delta t) < 0$ or $\alpha_i(t+\Delta t) > 1$ **then**
    Adjust ΔV_j using the redistribution procedure in Section 4.1.2.4
  end if
end for

4.1.2.1 Estimating the interface motion during $[t, t + \Delta t]$

To estimate the interface motion, the first step is to find the geometric centre of the interface, $\mathbf{x}_s$. This is described as follows;

Algorithm 4.5: Finding interface centroid, $\mathbf{x}_s$

for all interface cells **do**

1) Calculate the average point, x_N between the plane-face intersect points.

2) Decompose the interface into N triangles, with x_N as the common vertex, similar to Algorithm 4.2.

3) Find areas and geometric centres of the N triangles.

4) $\mathbf{x}_s$ is the area-weighted average of the N triangles.

end for

The velocity $\mathbf{u}_i(t)$ is interpolated to $\mathbf{x}_s$ using the OpenFOAM $interpolationCellPoint$ utility, where first, the cell is decomposed into tetrahedra. The tetrahedron containing $\mathbf{x}_s$ is identified and the velocity field is interpolated to its vertices. The velocity vector $\mathbf{U}_s$ at $\mathbf{x}_s$ is found by linearly interpolating the velocity from the tetrahedron vertices with inverse distance weights. The interface motion which is normal to itself is now $U_s = \mathbf{U}_S \cdot \mathbf{n}_\phi$. The direction of U_S indicates whether the cell is filling up of fluid L, or emptying out i.e. $U_S > 0$ indicates a cell filling up with fluid L.

4.1.2.2 Evolving the face-interface intersection line

The face-interface intersection line (FIIL) (see Fig. 4.12(a)) is evolved by estimating the times at which the interface reaches the vertices of face j, as it is translated along its normal at velocity U_s. The times at which the FIIL passes the vertices of face j ($\mathbf{X}_1$, ... , $\mathbf{X}_N$) is

$$t_k \approx t + (\mathbf{X}_k - \mathbf{x}_s) \cdot \frac{\mathbf{n}_\phi}{U_S}, \text{for } k = 1, ..., N \tag{4.17}$$

In this section, a quadrilateral section on a cell face swept by the movement of the FIIL is denoted with vertices E, F, G, H. In Fig. 4.12, the grey quadrilateral E, F, G, H is the area swept as the FIIL moves from t_4 to t_5. Therefore at the intermediate time $\tau \in [t, t + \Delta t]$, the locations $\tilde{H}(\tau)$ and $\tilde{G}(\tau)$ can be represented as follows;

$$\tilde{H}(\tau) = E + \frac{\tau - t_k}{t_{k+1} - t_k}(H - E),\ \tilde{G}(\tau) = F + \frac{\tau - t_k}{t_{k+1} - t_k}(G - F). \tag{4.18}$$

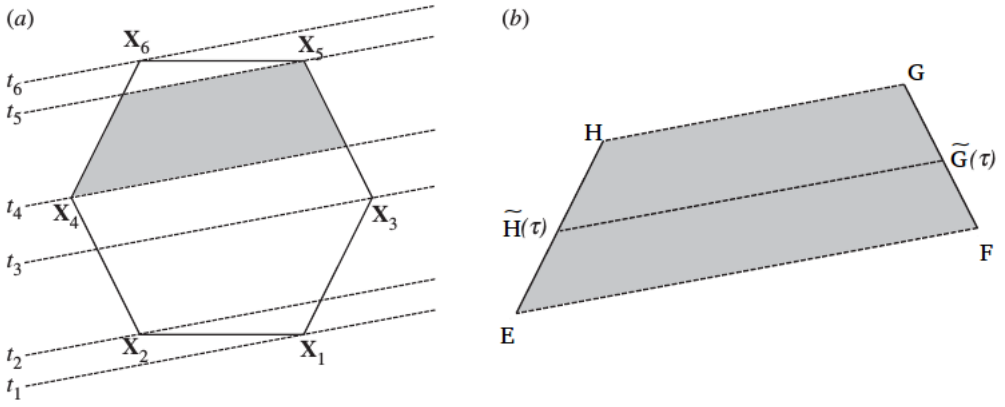


Figure 4.12: (a) The FIIL as it passes each face vertex. Take for example the trapezoid that resulted from the FIIL movement from t_4 to t_5, where it is denoted as E, F, G, H in (b). The intermediate positions as it moves from from EF to GH is denoted as $\tilde{H}(\tau)$ and $\tilde{G}(\tau)$. Graphics modified from Roenby et al. [2].

4.1.2.3 Time integral of submerged face area

Recall the instantaneous submerged area, $A_j(\tau)$ from Eq. 2.35. The times t_k obtained from the previous section are sorted and placed into a new list where $\tilde{t}_1, ..., \tilde{t}_M$, where $\tilde{t}_1 = t$ and $\tilde{t}_M = t + \Delta t$, such that $t < t_k < t + \Delta t$. The time integral in Eq. 2.35 is split as

$$\int_t^{t+\Delta t} A_j(\tau)d\tau = \sum_{k=1}^{M-1} \int_{\tilde{t}_k}^{\tilde{t}_{k+1}} A_j(\tau)d\tau, \tag{4.19}$$

where each k represents each quadrilateral interval as in Fig. 4.12 (b) . The submerged area at intermediate time $\tilde{t}_k \leq \tau \leq \tilde{t}_{k+1}$ can be expressed as

$$A_j(\tau) = A_j(\tilde{t}_k) + \frac{1}{2}\,\text{sgn}\,(U_S)\,|E\tilde{G}(\tau) \times F\tilde{H}(\tau)|. \tag{4.20}$$

From Eq. 4.18, this can be turned into the polynomial

$$A_j\tau = A_j(\tilde{t}_k) + P_k(\tau)^2 + Q_k\tau \tag{4.21}$$

where P_k and Q_k are coefficients found analytically from $E, F, \tilde{G}, \tilde{H}$. Obtaining these coefficients, the contribution to the time integral in Eq. 2.35 from the sub-interval $[\tilde{t}_k, \tilde{t}_{k+1}]$ is

$$\int_{\tilde{t}_k}^{\tilde{t}_{k+1}} A_j(\tau)d\tau = \frac{1}{3}P_k[\tilde{t}_{k+1}^3 - \tilde{t}_k^3] + \frac{1}{2}Q_k[\tilde{t}_{k+1}^2 - \tilde{t}_k^2] + A_j(\tilde{t}_k)[\tilde{t}_{k+1} - \tilde{t}_k]. \tag{4.22}$$

All these sub-contributions are added up as in Eq. 4.19. This is inserted into Eq. 4.16 to obtain $\Delta V_j(t, t + \Delta t)$. This procedure is repeated for all downwind faces of the interface cell to get

$$\alpha_i(t + \Delta t) = \alpha_i(t) - \frac{1}{V_i}\sum_{j \in B_i} s_{ij}\Delta V_j(t, t + \Delta t) \tag{4.23}$$

which updates the volume fraction α_i for the new time step in the cell. On non-downwind faces, ΔV_j is simply set as

$$\Delta V_j = \alpha_{upwind}\mathbf{F}_j\Delta t. \tag{4.24}$$

4.1.2.4 Bounding procedure

Roenby et al. cautioned that this method of advection may produce some boundedness issues. Namely, it might cause the updated volume fractions to be slightly outside its meaningful range of $0 \leq \alpha_i \leq 1$. This is believed to occasionally happen when a cell is being completely emptied or filled, and the algorithm may cause an over-filling or over-emptying. Simply truncating the volume fractions such that

$$\alpha_i = \begin{cases} 0, & \text{if } \alpha_i < 0 \\ 1, & \text{if } \alpha_i > 0 \end{cases} \tag{4.25}$$

will disrupt the volume conservation even if α_i missed 0 or 1 by a very small amount. Therefore, Roenby et al. addressed this issue by proposing a fluid redistribution procedure, which is also employed in this work.

For cells to exceed their maximum allowed volume fraction ($\alpha_i > 1$), it is likely they are just upwind of the interface i.e. the interface is moving into fluid G. Therefore cells upwind of it are likely to also be completely filled, and unable to receive any more fluid L. Therefore any extra fluid in cell C_i can be passed on through its downwind faces.

This requires one to distribute the extra fluid amongst these downwind faces. If the total extra fluid volume in cell C_i is denoted as V_{extra}, where

$$V_{extra} = (\alpha_i - 1)V_i \tag{4.26}$$

where V_i is the volume of the mesh cell, and the cell has N_D downwind faces, the fluid is distributed for each of its downwind face j, as

$$V_{j+} = V_{extra}\frac{|\mathbf{F}_j|}{\sum_{d=1}^{N_D}\mathbf{F}_d} \tag{4.27}$$

where V_{j+} is the amount of extra fluid distributed through face j, and $\sum_{d}^{N_D}\mathbf{F}_D$ is the sum of the fluxes of all cell C_i's downwind faces. The maximum amount of fluid L passing through a face j is limited to a maximum of $\mathbf{F}_j\Delta t$, so

$$V_{j+} = min(V_{j+}, \mathbf{F}_j\Delta t - \Delta V_j). \tag{4.28}$$

A new value of $\Delta V_j(t, t+\Delta t)$ is obtained, which gives a new value of $\alpha_i(t+\Delta t)$ in Eq. (4.23). This new value of α_i is then checked if it still exceeds 1, in which case the distribution is repeated for the remaining downwind faces until all the surplus has been redistributed.

If the cells have undershot their minimum volume fraction ($\alpha_i < 0$), the same procedure as described above also applies, but from the perspective of fluid G whose volume fraction is represented by $\beta_i = 1 - \alpha_i$. This gives $\beta_i > 1$ when $\alpha_i < 0$. In this case, the volume of fluid G transported across face j between $[t, t+\Delta t]$ is $\Delta\tilde{V}_j \equiv \mathbf{F}_J\Delta t - \Delta V_j$.

The redistribution procedure then corrects the $\Delta\tilde{V}_j$ values, and one can calculate the new $\Delta V_j = \mathbf{F}_j \Delta t - \Delta\tilde{V}_j$, which is then used to update $\alpha_i(t + \Delta t)$ in Eq. (4.23).

4.1.3 Advection and reinitialisation of the LS function

Subsection 4.1.2 has addressed the advection of the VOF field. The advection of the LS field is much simpler; as in [108], a first order upwind method is used to solve

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0. \tag{4.29}$$

Advecting the LS field introduces disruption and no longer renders it a signed distance function from the interface. This is a known issue of the LS method and is remedied by reinitialising it after each time step. Firstly this requires that the plane equation coefficients A_i, B_i, C_i, D_i be found using the advected value of ϕ, which is $\phi*$, and adjusted such that the volume cut by the plane matches the volume given by $\alpha_i(t+\Delta t)$. This procedure has been explained in Section 4.1.1.

Afterwards, for each neighbouring cell within up to five layers away from the interface cell, their exact distance to the nearest interface is computed to obtain the value of ϕ at the new time step. Some methods have been described in [105], [45], [44], [44], [109], [46]. Most of these deal with simple configurations/uniform hexahedral meshes. Since this work is in three dimensions, many complications arise, mainly from the fact that the interface could be oriented in numerous ways, and the computation of the location of a vertex for a fluid volume can prove difficult.

In this work, this issue has been pre-addressed by the usage of lists of plane/cell cutting points as described in the cell decomposition section. Therefore, the fluid vertices in interface cells (the intersect between plane and cell) is already available for use.

A further point to be found is the projection point of the neighbour cell centres onto the interface, found as detailed by [110]. Using the available data, the exact distance from neighbouring cell centres to the interface is calculated. This calculation is performed

for five layers of neighbour cells i.e. if a cell containing an interface is cell i, the closest signed distance is calculated for its immediate neighbours and the neighbours of these immediate neighbours.

This reinitialisation procedure is summarised as follows;

1. The LS and VOF fields have been advected, giving the VOF values for the new time step, and an intermediate value of LS, $\phi*$.

2. For interface cells, the plane is reconstructed so that the volume under the plane matches the value of the advected VOF. The value of D from this plane equation updates the LS values for the interface cells for the new time step.

3. To update the LS values in the local area of the interface, a list of neighbour cells are identified, which is a layer of five cells around the interface cell i. If the neighbour cell is also an interface cell, this cell is eliminated from the list as its LS value has already been updated.

4. A large numeric value d_{temp} is instantiated for the LS field in this neighbour list for all the interface cells, maintaining the sign of the LS value.

5. For all cells in the neighbour list of the first interface cell i, the following values are calculated:

   (a) The distances of the cell centre of the neighbour cell, to the fluid vertices of the interface cell. The minimum value is stored as d_v.

   (b) The distance of the cell centre of the neighbour cell, to its projection point on the interface plane in the interface cell i. This value is stored as d_c, only if the projection point lies within the boundaries of the interface cell i.

   (c) The distances of the cell centre of the neighbour cell, to the face centroids of the interface cell faces, if the sign of the LS function at that face centroid is different to the sign of the neighbour cell. The minimum value is stored as d_f.

(d) The minimum of d_{temp}, d_v, d_c, and d_f is stored as the new d_{temp}.

6. The process is repeated for the neighbours of the next interface cell, and d_{temp} value in the neighbour cell is updated if a new shorter distance to an interface cell is found. When the process is performed for all the neighbour cells of all the interface cells, the final d_{temp} in each neighbour cell is the updated value of LS at the new time step.

4.2 Validation on structured and unstructured grids

This section presents some numerical verifications of the developed CLSVOF scheme that was implemented into OpenFOAM-2.1.x, compared against interFoam which is OpenFOAM's native VOF-based two-phase incompressible solver, and interFlow, a geometric VOF-based OpenFOAM solver using isoAdvector as its advection solver.

4.2.1 3D advection of a sphere

To validate 3D interface capturing capabilities, a sphere of radius $r = 0.25$ is transported in a uniform flow across a rectangular domain of [4,1,1] m with a constant uniform velocity of (1, 0, 0) ms^{-1}. The sphere is initially centred at (0.5, 0.5, 0.5) as in Fig. 4.13 and the simulation is run to T = 3. The exact solution of the final position of the sphere centre is (3.5, 0, 0). The test is performed under six different mesh conditions, detailed in Table 4.1.

Table 4.1: Mesh parameters of the 3D advection test

Mesh	N	Δt
Structured hexahedral	16384	4×10^{-3}
	131072	2×10^{-3}
	1048576	1×10^{-3}
Unstructured tetrahedral	10082	1×10^{-3}
	32725	5×10^{-4}
	138090	2.5×10^{-4}

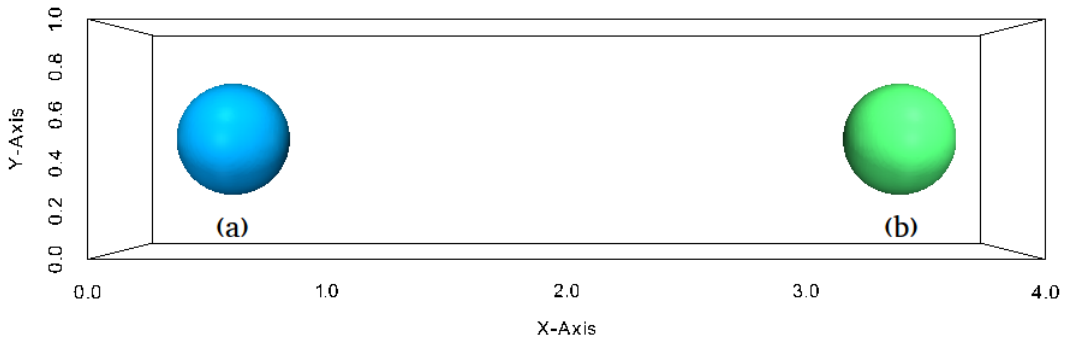


Figure 4.13: Initial position of the sphere in the 3D advection test (a) and its expected final position (b).

The $L1$ errors are calculated as

$$L1 = \frac{\sum_N |\alpha_{exact} - \alpha_i(t)|}{N} \tag{4.30}$$

where α_{exact} is the exact solution, $\alpha_i(t)$ is the calculated solution, and N is the number of mesh cells.

CLSVOF consistently produces the smallest L1 errors in all the test parameters, as seen in Table 4.2 and Table 4.3 . The results using CLSVOF display the least amount of distortion in the final shape, and this difference could be seen even in the fine mesh

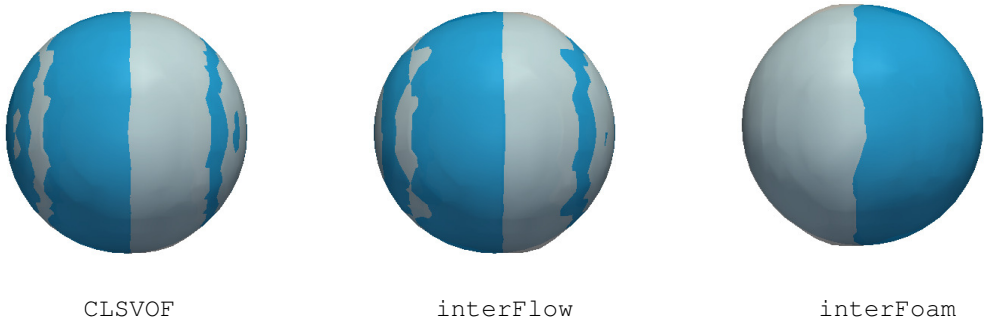


Figure 4.14: Position of advected 3D sphere using hexahedral mesh (N=131072) at $t = 3$. Grey sphere is the VOF=0.5 contour and blue sphere is exact solution.

test as in Fig. 4.14, where the final position of the sphere is compared to the exact solution (denoted in blue in the coloured version). In Fig. 4.15, it is again seen that CLSVOF is the least distorted, especially visible when compared to interFoam. Figures 4.16, 4.17 show that error naturally decreases as mesh is refined.

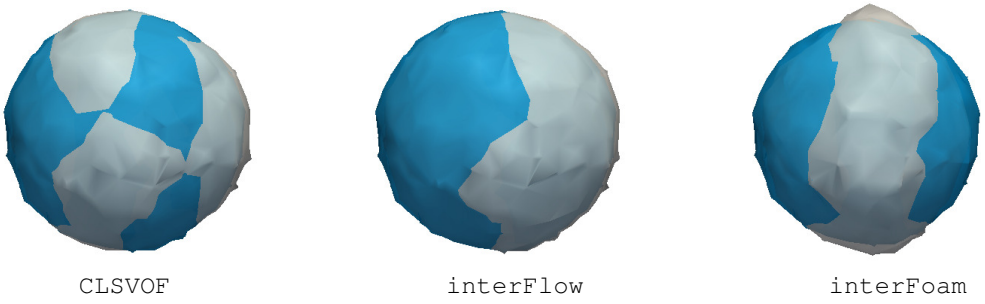


Figure 4.15: Position of advected 3D sphere using tetrahedral mesh (N=138090) at $t = 3$. Grey sphere is the VOF=0.5 contour and blue sphere is exact solution.

The volume conservation properties of each method is measured using the volume deviation, which is the relative change in the total volume of fluid L in the domain

compared to its initial volume;

$$\delta V = \frac{\sum_i \alpha_i(t)V_i - \sum_i \alpha_i(0)V_i}{\sum_i \alpha_i(0)V_i} \times 100, \tag{4.31}$$

where $\sum_i \alpha_i(0)V_i$ is the total volume of fluid L at $t = 0$. Tables 4.4 and 4.5 show the volume deviation errors for the hexahedral mesh test. It can be seen that CLSVOF and interFlow has comparable volume preservation, with both being in the sufficiently small range of $2 \times 10^{-9}\%$ and CLSVOF having a slight edge. However, in this regard it is found that interFoam performs better than either CLSVOF or interFlow for volume conservation.

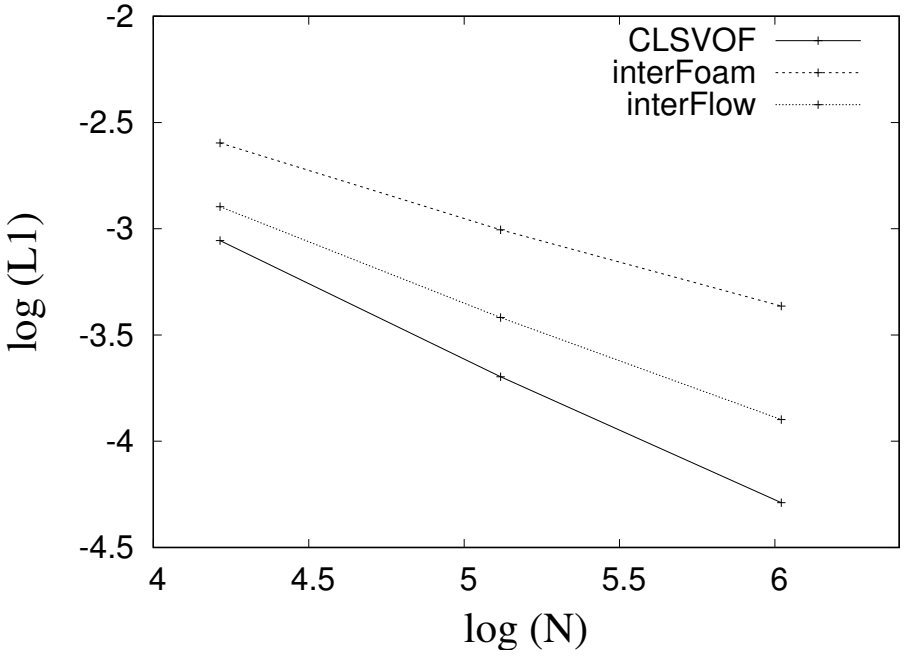


Figure 4.16: L1 error for the advected 3D sphere on successively refined hexahedral mesh, where N is the mesh number.

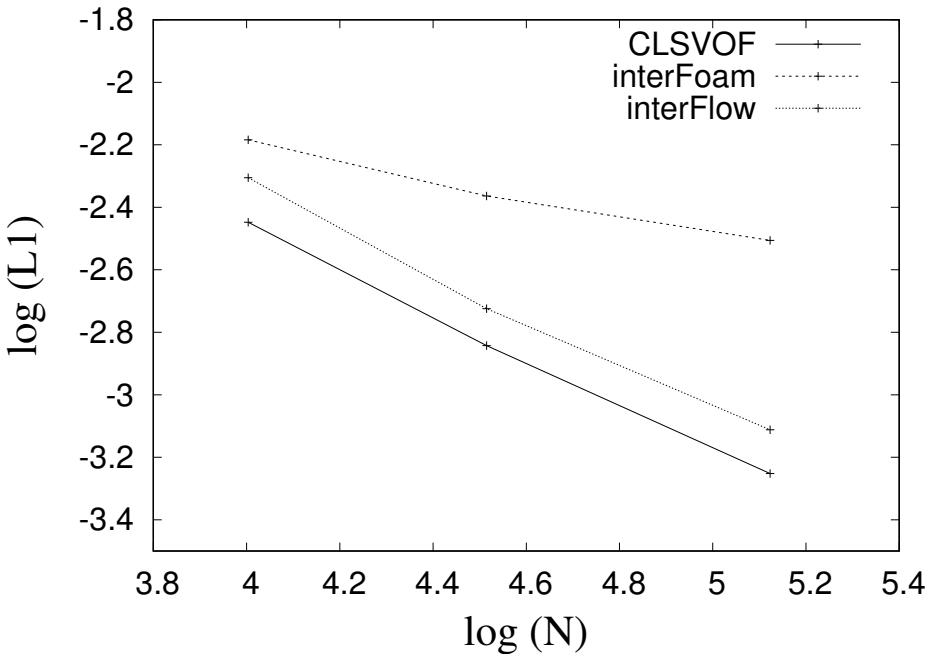


Figure 4.17: L1 error for the advected 3D sphere on successively refined tetrahedral mesh, where N is the mesh number.

Table 4.2: L_1 errors in 3D advection test using structured mesh

N	CLSVOF	interFlow	interFoam
16,384	8.79×10^{-4}	1.27×10^{-3}	2.53×10^{-3}
131,072	2.01×10^{-4}	3.82×10^{-4}	9.89×10^{-4}
1,048,576	5.14×10^{-5}	1.26×10^{-4}	4.33×10^{-4}

Table 4.3: L_1 errors in 3D advection test using unstructured mesh

N	CLSVOF	interFlow	interFoam
10,082	3.57×10^{-3}	4.95×10^{-3}	6.54×10^{-3}
32,725	1.43×10^{-4}	1.88×10^{-4}	4.32×10^{-3}
138,090	5.60×10^{-4}	7.73×10^{-4}	3.12×10^{-3}

Table 4.4: Percentage volume deviation δV errors in 3D advection test using structured mesh.

N	CLSVOF	interFlow	interFoam
16,384	-1.7×10^{-9}	-2.0×10^{-9}	-3×10^{-12}
131,072	-1.9×10^{-9}	-2.3×10^{-9}	-4.8×10^{-12}
1,048,576	-1.8×10^{-9}	-1.9×10^{-9}	-2.1×10^{-12}

Table 4.5: Percentage volume deviation δV errors in 3D advection test using unstructured mesh.

N	CLSVOF	interFlow	interFoam
10,082	-1.6×10^{-7}	-1.3×10^{-7}	-7.6×10^{-12}
32,725	-1.2×10^{-7}	-1.1×10^{-7}	-5.0×10^{-11}
138,090	-8.3×10^{-8}	-8.4×10^{-8}	-2.7×10^{-11}

In this test the simulations are performed on an Intel Core i7-4790K machine, with 4 x 4.00GHz CPUs in parallel. The time taken for each simulation is listed as follows;

Table 4.6: Simulation time for the 3D advection test for CLSVOF, interFlow, and interFoam.

		Simulation time (s)		
Mesh	**N**	**CLSVOF**	**interFlow**	**interFoam**
Structured	16384	11	4	9
	131072	213	74	139
	1048576	2040	790	1440
Unstructured	10082	13	7	9
	32725	124	42	54
	138,090	1211	313	611

Table 4.6 shows that interFlow is the fastest scheme, taking the least amount of time for all three methods tested. CLSVOF takes about twice the time it takes for the interFoam method. The speed penalty taken by the CLSVOF method is possibly due to the complex cell manipulation as well as the iteration for finding the correct plane.

4.2.2 Rotation of Zalesak's disc

The developed scheme is tested with a two dimensional solid body rotation as proposed by Zalesak [111], later used by Rudman [112] and many other researchers. A slotted disk as indicated in Figure 4.18 is rotated in a constant vorticity field given by the stream function

$$\psi(x,y) = -\frac{\omega}{2}[(x-x_O)^2 - (y-y_O)^2] \tag{4.32}$$

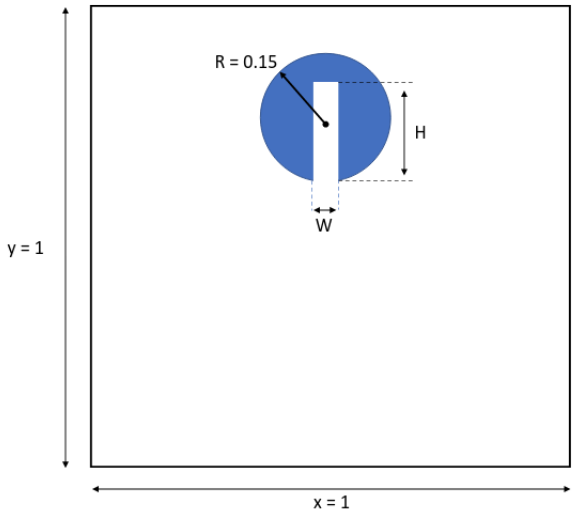


Figure 4.18: Schematic representation of the Zalesak problem at $t = 0$, where the disk is centred at $(0.0, 0.25)$, $H = 0.25$, and $W = 0.05$.

where $O = (x_O, y_O)$ is the centre of rotation and ω is the constant angular velocity. After a revolution 2π the disk returns to its initial position. The test is performed on a 100×100 grid and 200×200 grid. The results are presented using the 0.5-contour of the α field.

Results from this test show that interFlow and CLSVOF have smaller $L1$ errors than interFoam for both structured and unstructured meshes in both resolutions. InterFoam results display a distorted interface, to a considerably worse degree in the coarser unstructured mesh case. However, the results obtained by interFlow and CLSVOF are almost similar, with interFlow having a slightly smaller error in this case.

Table 4.7: L1 errors in structured 2D Zalesak test.

N	CLSVOF	interFlow	interFoam
10000	4.15×10^{-3}	3.88×10^{-3}	7.46×10^{-3}
40000	1.81×10^{-3}	1.80×10^{-3}	3.69×10^{-2}

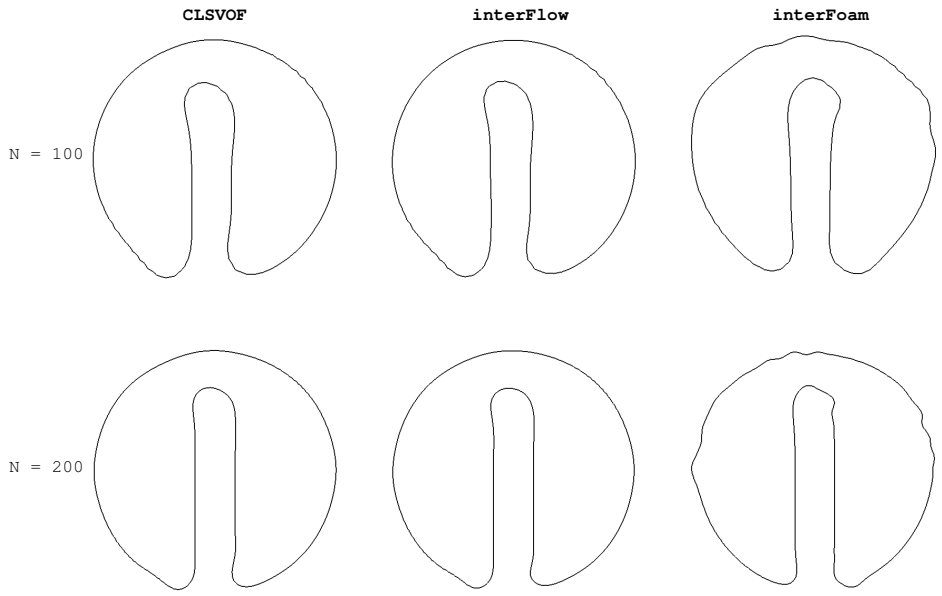


Figure 4.19: Results of the Zalesak test on structured quadrilateral mesh after 1 rotation.

Table 4.8: L1 errors in unstructured 2D Zalesak test.

N	CLSVOF	interFlow	interFoam
10472	4.65×10^{-3}	4.15×10^{-3}	7.59×10^{-3}
39362	2.40×10^{-3}	2.30×10^{-3}	3.99×10^{-2}

4.2.3 Vortex deformation transport test in 2D

As per Rider and Kothe [113] as well as Menard [43], we assess the proposed scheme with a non-uniform vorticity field using the initial setup in Fig. 4.21. The time-dependent velocity field of the problem is given by the stream function

$$\Psi(x, y, t) = \frac{1}{\pi} \sin^2(\pi x) \sin^3(\pi y) \cos(\frac{\pi t}{T}). \tag{4.33}$$

The velocity field (4.33) stretches the initial circle into a filament spiralling towards the centre of the domain. This stretching may cause the material to tear if the mesh

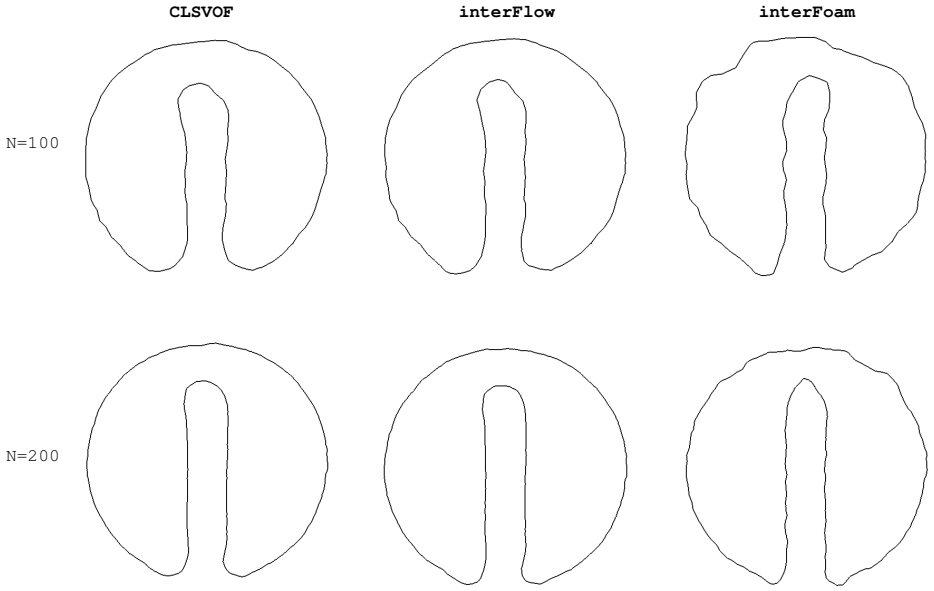


Figure 4.20: Results of the Zalesak test on unstructured triangle mesh after 1 rotation.

Parameter	Value
domain	[0,1] x [0,1]
circle	C = (0.5, 0.25), R = 0.15
period	$T_1 = 8$, $T_2 = 4$

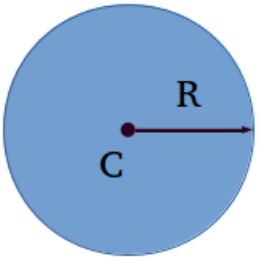


Figure 4.21: Initial setup of the Rider-Kothe vortex deformation test

resolution is insufficient or the interface tracking method is not truly robust. The exact solution should see it restored to its initial shape upon reaching $t = T$. This test is a reasonable challenge for interface capturing methods as the liquid ligament will be stretched relatively thin compared to the grid size. At $t = T/2$ the velocity field is

reversed such that at $t = T$ the circular shape and its initial location is restored.

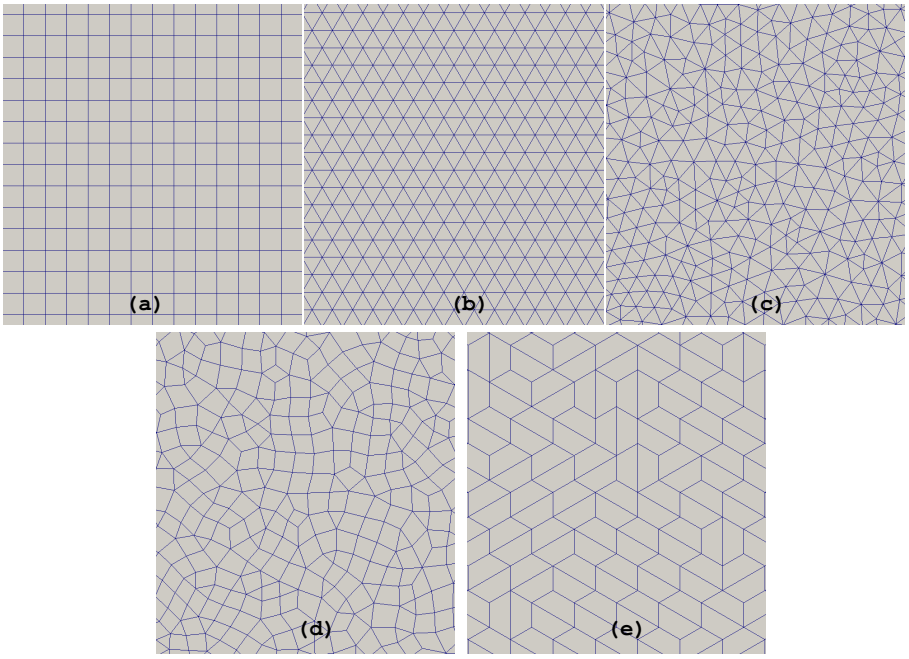


Figure 4.22: Meshes used for the 2D vortex deformation test, where (a) structured quadrilateral, (b) structured triangular, (c) unstructured triangular, (d) unstructured quadrilateral, and (e) unstructured polygonal .

Table 4.9: Mesh conditions for the 2D vortex deformation test

Mesh type	N	CFL number
Structured quadrilateral	10000	0.1
Structured triangular	10484	0.05
Unstructured triangular	26548	0.05
Unstructured quadrilateral	14961	0.1
Unstructured polygonal	5377	0.1

The test is done using a long cycle (T_1) and a short cycle (T_2). The velocity field is reversed at T=4 and T=2 respectively, so the T_1 test is expected to be more challenging as the deformation will be more severe. We compare against interFoam and interFlow

on five different types of meshes as shown in Fig. 4.22, which is a combination of structured and unstructured meshes of various types of polyhedra.

Fig. 4.23 for hexahedral meshes which shows the simulation as it stretches to its maximum at $T = 4$, demonstrates how CLSVOF produced the least surface breakup at the tail end of the filament. interFlow follows behind, and interFoam displays a very diffusive interface region with tail breakup starting at the 7 o'clock position; far more extensive breakup than either interFlow or CLSVOF. The same can said for $T = 6$. Fig. 4.23 also shows that at $T = 8$ where the solution returns to its initial position, CLSVOF has maintained the best circularity, followed by interFlow and interFoam.

This trend continues for results obtained for structured and unstructured triangular meshes in Figures 4.24, 4.25, and unstructured quadrilateral mesh in Fig. 4.26 . interFoam performed particularly poorly on unstructured triangular meshes as can be seen in Fig. 4.25, showing a large amount of interface breakup during maximum stretch and failing to recover the initial shape at $T = 8$. On the unstructured polygonal mesh in Fig. 4.27, all three schemes produced solutions that trail on the left-hand side, with interFoam displaying it to a severe degree.

It can be seen that CLSVOF is consistently able to resolve the thin filaments with minimal tail-breakup at maximum stretch, which occurs to a higher degree in interFlow as in Figures (4.23, 4.24). interFoam proves to be diffusive across all mesh parameters and performs poorly on unstructured meshes as seen in Figures (4.25, 4.27). The errors are calculated as follows;

$$E_{L1} = \frac{|\alpha^{init} - \alpha^{final}|}{N} \tag{4.34}$$

where α^{init} is the initial field values of the VOF field, α^{final} is the field values at time T, and N is the mesh size. The results are displayed in Table 4.10 and shows that CLSVOF has the least errors in all test parameters.

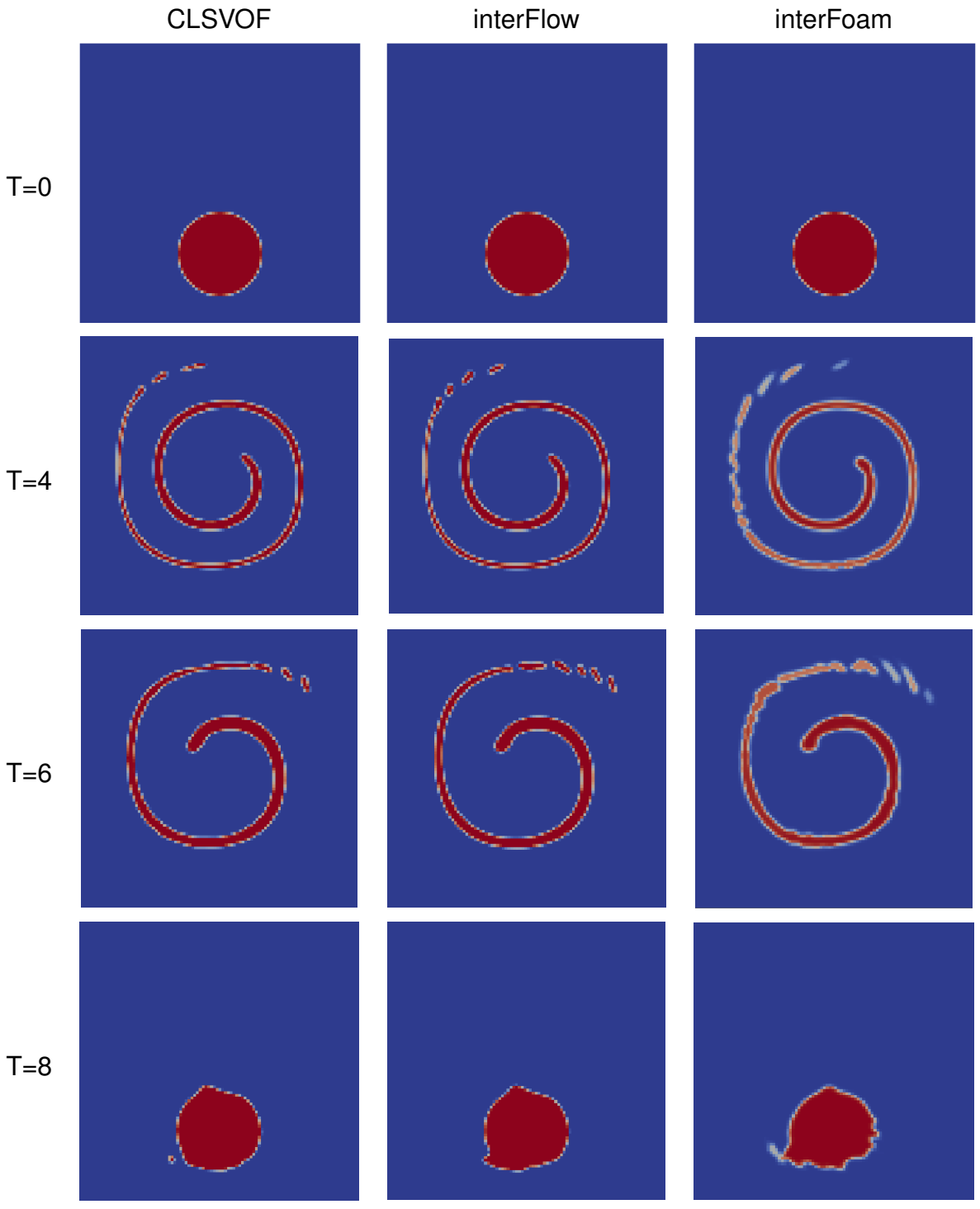


Figure 4.23: 2D vortex deformation test results for CLSVOF, interFlow, and interFoam on a structured quadrilateral mesh, at times T=4, 6, 8.

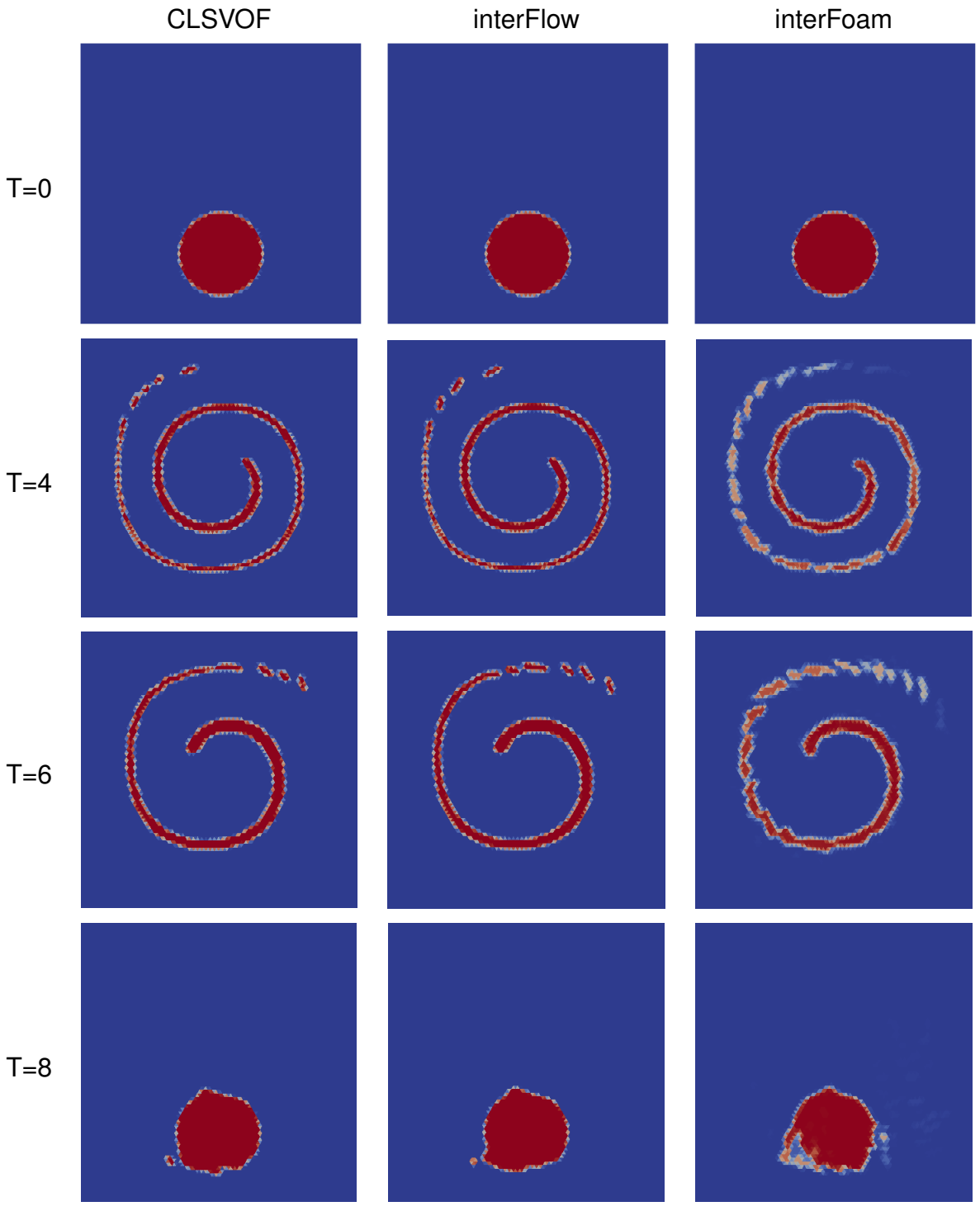


Figure 4.24: 2D vortex deformation test results for CLSVOF, interFlow, and interFoam on a structured triangular mesh, at times T=4, 6, 8.

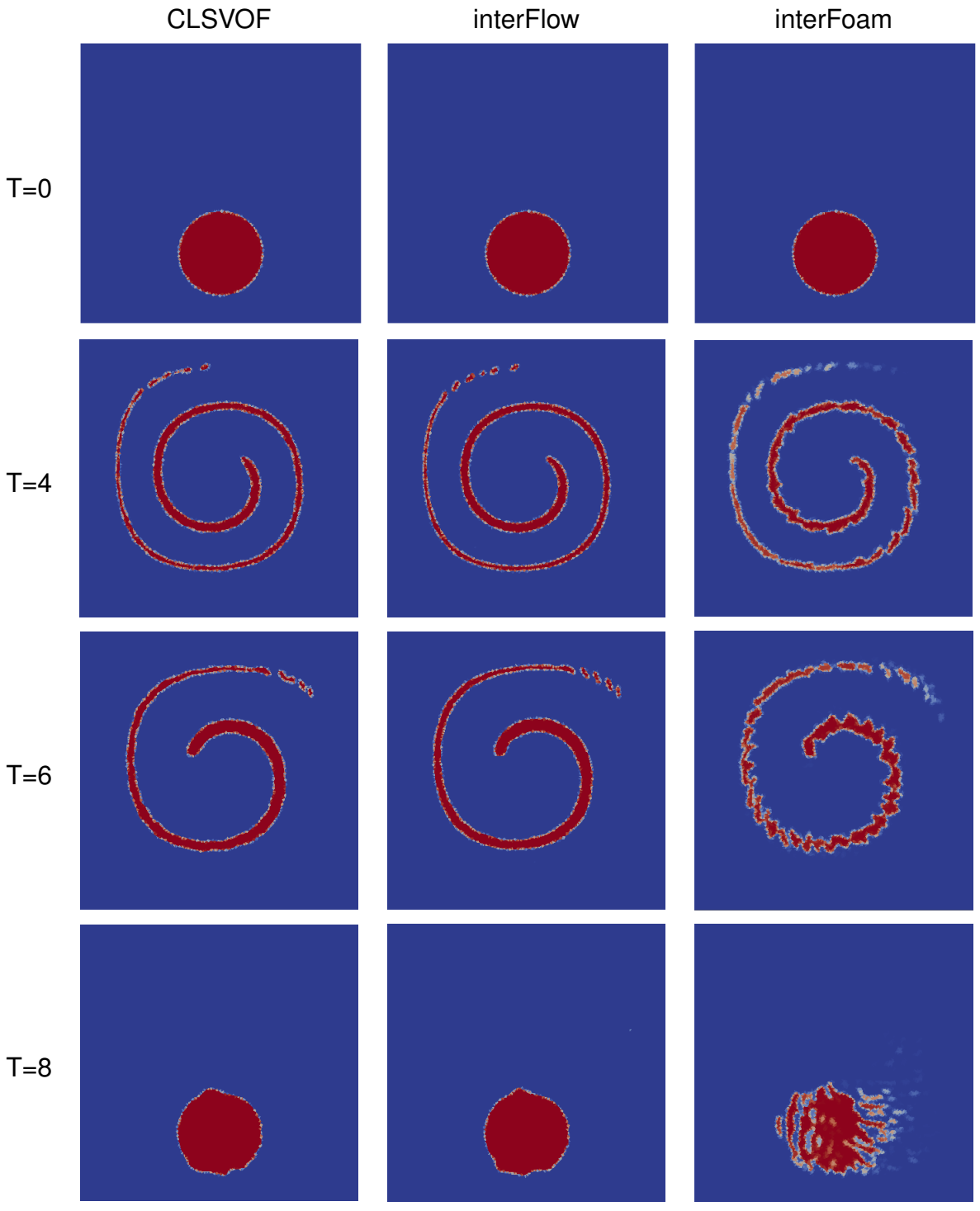


Figure 4.25: 2D vortex deformation test results for CLSVOF, interFlow, and interFoam on an unstructured triangular mesh, at times T=4, 6, 8.

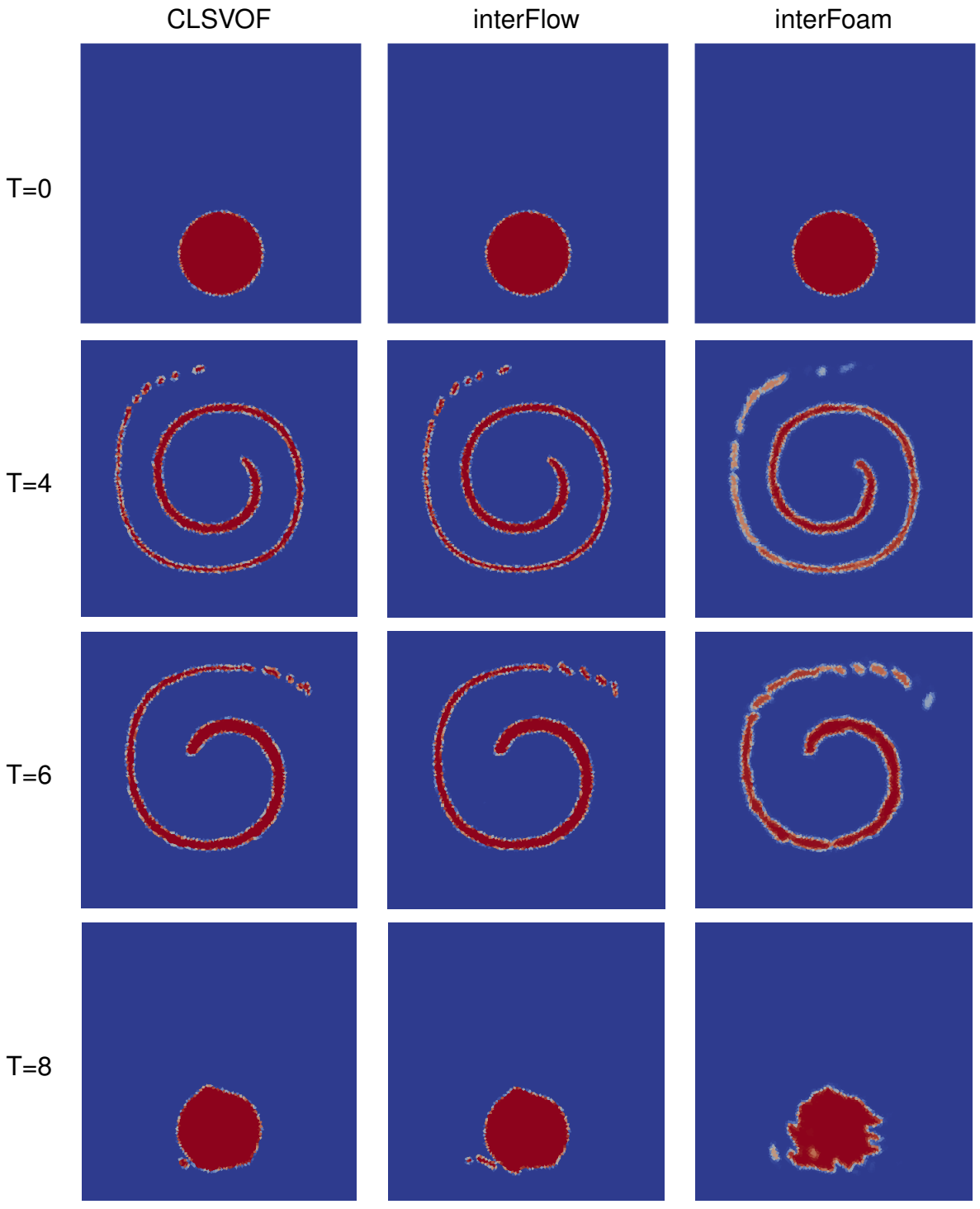


Figure 4.26: 2D vortex deformation test results for CLSVOF, interFlow, and interFoam on an unstructured quadrilateral mesh, at times T=4, 6, 8.

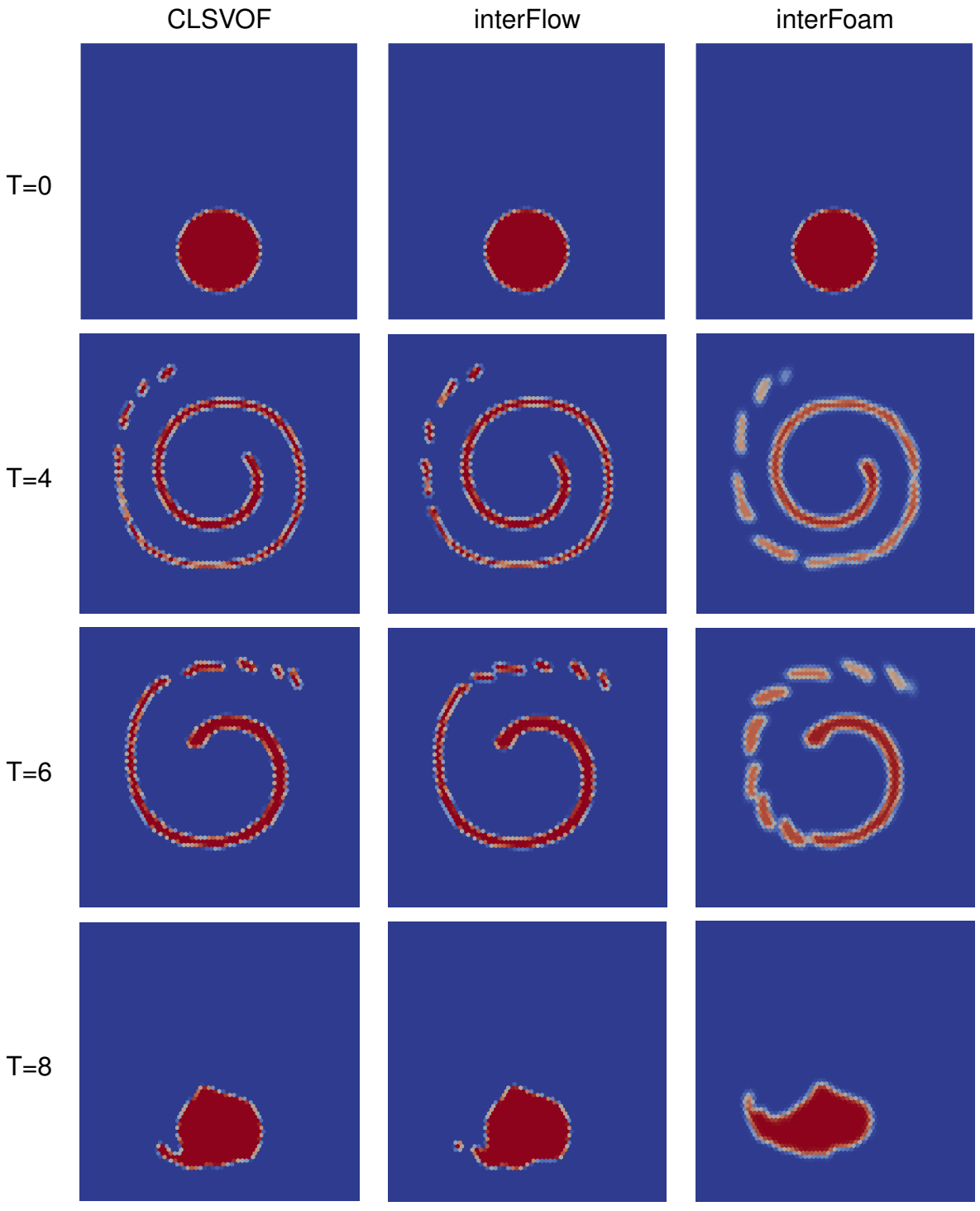


Figure 4.27: 2D vortex deformation test results for CLSVOF, interFlow, and interFoam on an unstructured polygonal mesh, at times T=4, 6, 8.

Table 4.10: L1 errors in 2D Rider-Kothe test for T_1

Mesh	CLSVOF	interFlow	interFoam
Structured quadrilateral	3.84×10^{-3}	5.74×10^{-3}	1.23×10^{-2}
Structured triangular	5.36×10^{-3}	6.57×10^{-3}	1.26×10^{-2}
Unstructured triangular	2.91×10^{-3}	3.25×10^{-3}	3.06×10^{-2}
Unstructured quadrilateral	4.29×10^{-3}	5.66×10^{-3}	1.20×10^{-2}
Unstructured polygonal	1.11×10^{-2}	1.18×10^{-2}	3.99×10^{-2}

For the T_2 variant of the test, a uniform quadrilateral mesh (N=10,000) and an unstructured triangular mesh (N=26,548) are used. Again CLSVOF is seen to perform the best both qualitatively (Figures 4.28 & 4.29) and quantitatively (Table 4.11), with slight improvements over interFlow and major improvements over interFlow in structured and unstructured meshes for interface capturing tests.

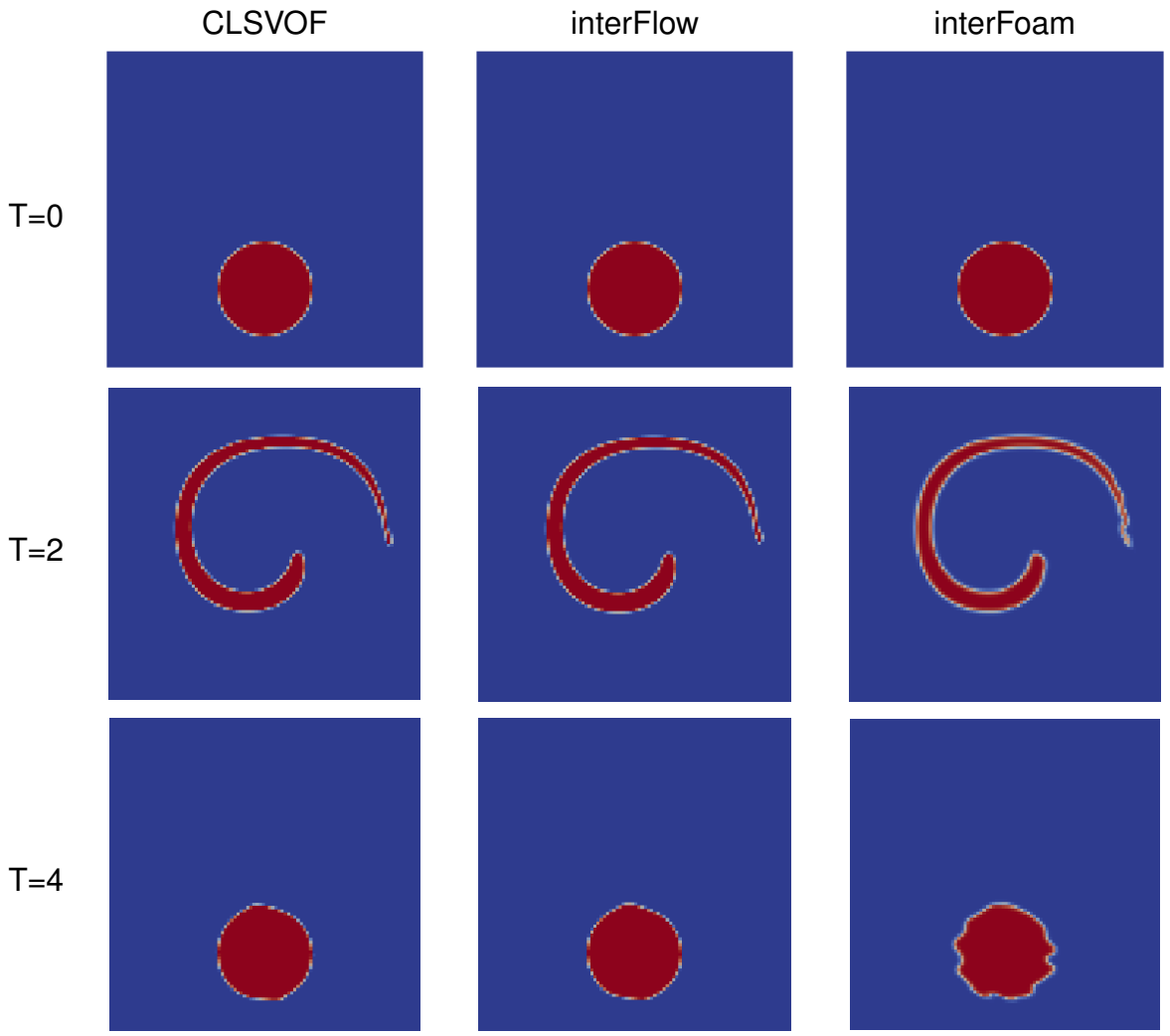


Figure 4.28: 2D vortex deformation test results for CLSVOF, interFlow, and interFoam on a structured hexahedral mesh, at times T=2, 4.

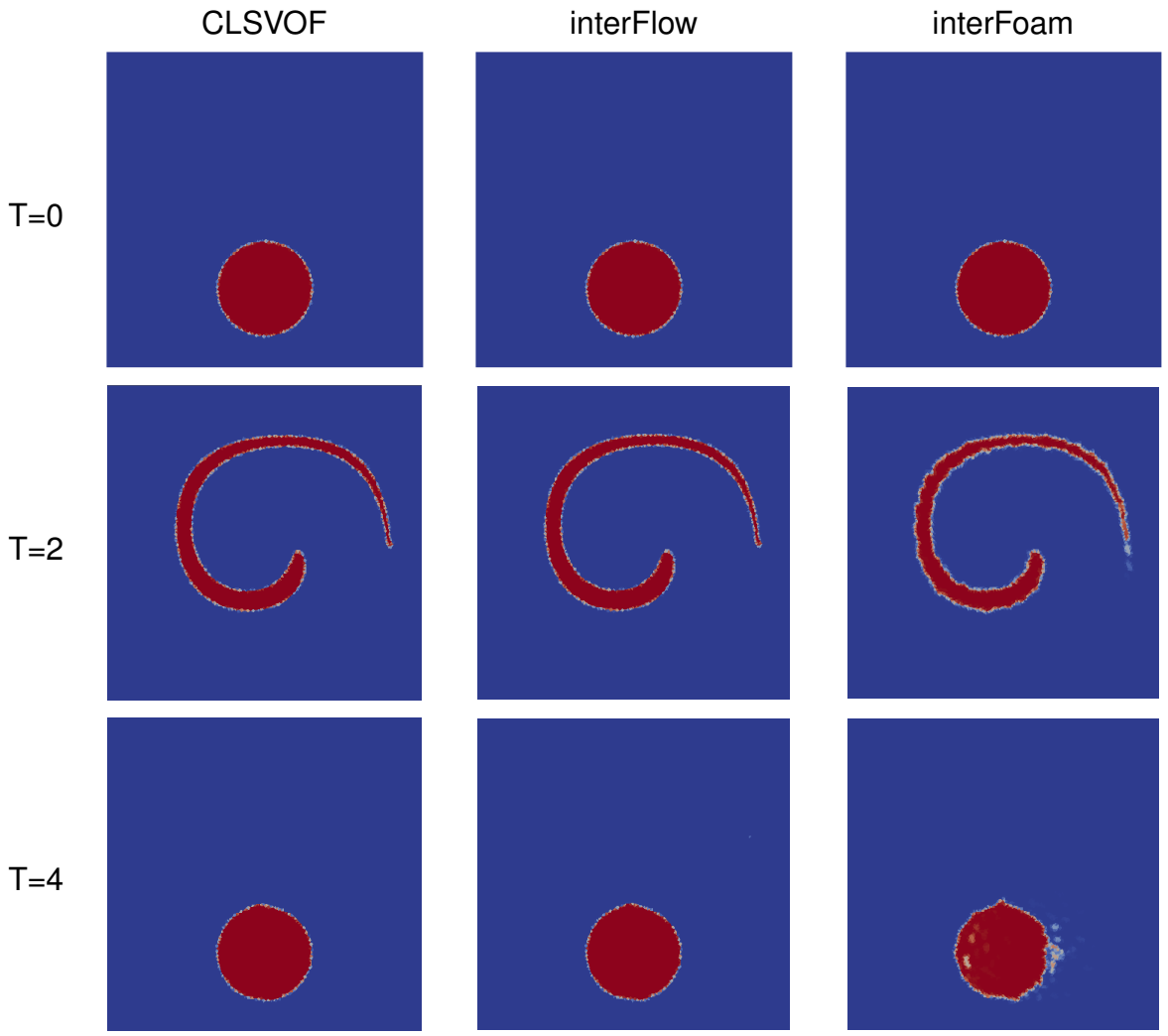


Figure 4.29: 2D vortex deformation test results for CLSVOF, interFlow, and interFoam on an unstructured tetrahedral mesh, at times T=2, 4.

Table 4.11: L1 errors in 2D Rider-Kothe test for T_2

Mesh	CLSVOF	interFlow	interFoam
Structured quadrilateral	1.36×10^{-3}	1.51×10^{-3}	5.17×10^{-2}
Unstructured triangular	1.32×10^{-3}	1.46×10^{-3}	5.48×10^{-2}

4.3 Conclusion

A new CLSVOF interface capturing scheme utilising the isoAdvector VOF-advector from interFlow was implemented in OpenFOAM-2.1.x. Detailed methodology for decomposing general convex polyhedra and reconstructing an interface in each multiphase cell are given.

The proposed CLSVOF scheme was validated for various structured and unstructured mesh types in 2D and 3D and was found to work well in the validation tests. It was compared against interFlow and interFoam. The CLSVOF scheme is found to consistently perform better than interFoam in all tests. In the Zalesak's rotation test, both the interFlow and CLSVOF schemes outperformed interFoam, with CLSVOF and interFlow displaying visually similar results. The CLSVOF scheme has retained the good mass conservation property of interFlow, with a volume deviation of only around $1 \times 10^{-9}\%$ for structured meshes and $1 \times 10^{-8}\%$ for unstructured meshes.

Chapter 5

Validation and Application of CLSVOF Solver with Surface Tension Implementation

Two-phase flow problems involving surface tension forces arise in many applications. A robust calculation for local curvature is essential for an accurate treatment of surface tension. In this chapter, the surface tension formulation used is described along with the implementation of the Yokoi dynamic contact angle model [7]. The proposed CLSVOF solver is validated across a range of tests, ranging from the simple static droplet test to the more intricate cases of droplet splashing and collision as well a liquid jet in a gaseous crossflow.

5.1 Surface tension formulation

The Navier-Stokes equation for two-phase flows is as follows;

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\cdot(\rho\mathbf{u}\mathbf{u}) = -\nabla p + \nabla\cdot(2\mu D) + \mathbf{F}_\sigma + \rho\mathbf{g} \tag{5.1}$$

where ρ is the density, $\mathbf{u}$ is the velocity vector, μ is the dynamic viscosity, $\mathbf{D}$ is the rate of deformation tensor, $\mathbf{g}$ is the gravity vector, and $\mathbf{F}_\sigma$ is the surface tension force as a body force. Unlike single phase flows, ρ and μ are discontinuous due to the presence

of a fluid interface. In this work, these are described as follows;

$$\rho = \rho_L \alpha + \rho_G (1 - \alpha), \tag{5.2}$$

$$\mu = \mu_L \alpha + \mu_G (1 - \alpha), \tag{5.3}$$

with L and G subscripts denoting the two phases Liquid and Gas respectively. This ensures the mean momentum of the contents of the cell are used in the interface cell, while cells away from the interface retain the correct liquid and gas properties.

The surface tension force $\mathbf{F}_\sigma$ in Eq. (5.1) is calculated based on the Continuum Surface Force (CSF) model as proposed by Brackbill et al. [60],

$$\mathbf{F}_\sigma = \sigma \mathcal{K} \nabla \alpha, \tag{5.4}$$

where σ is the surface tension coefficient of the liquid in gas and $\mathcal{K}$ is the mean interface curvature. The usage of the gradient of the α field in this equation ensures the surface tension is limited to the region of the interface. The calculation of curvature $\mathcal{K}$ in the interFoam solver is found as follows

$$\mathcal{K} = -\nabla \cdot \frac{\nabla \alpha}{|\nabla \alpha|}. \tag{5.5}$$

In this implementation of CLSVOF, the curvature is found using the smoother LS field,

$$\mathcal{K} = -\nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}. \tag{5.6}$$

which should lead to a more accurate estimate of the surface tension force.

5.2 Dynamic contact angle formulation

The contact angle of a fluid is the angle where the liquid-vapour interface meets a solid surface. This contact angle is unique for a system of solid, liquid, and vapour at a certain temperature and pressure. A higher contact angle signifies a less-wettable

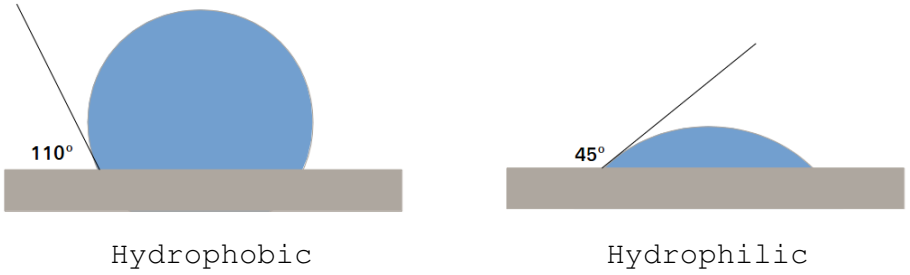


Figure 5.1: Different contact angles on a surface

fluid, and vice versa for a lower contact angle. Specifically, if the liquid contact angle is greater than $90\,\mathrm{deg}$ it is considered hydrophobic. If it is less than $90\,\mathrm{deg}$ it is considered hydrophilic [114].

OpenFOAM enforces the contact angle between the liquid/gas interface and the wall by correcting the local interface normal vector on a face f on a wall boundary. Let the current normal vector of the fluid interface at the wall be $\hat{\mathbf{n}}_{\phi,0}$. The normal vector to the wall is denoted $\mathbf{n}_{wall}$. The current contact angle between $\hat{\mathbf{n}}_{\phi,0}$ and $\mathbf{n}_{wall}$ is θ_0 and is calculated as

$$\cos\theta_0 = \hat{\mathbf{n}}_{\phi,0} \cdot \mathbf{n}_{wall}. \tag{5.7}$$

The interface normal at the wall needs to be corrected to the target normal $\mathbf{n}_\phi$ using the target contact angle θ, which is a set parameter for each simulation. This target normal must fulfil the two following criteria;

The angle between $\mathbf{n}_\phi$ and $\mathbf{n}_{wall}$ must fulfil

$$\cos\theta = \hat{\mathbf{n}}_\phi \cdot \mathbf{n}_{wall}, \tag{5.8}$$

and $\mathbf{n}_\phi$ must lie in the plane spanned by $\hat{\mathbf{n}}_{\phi,0}$ and $\mathbf{n}_{wall}$, such that

$$\mathbf{n}_\phi = A_{ca}\hat{\mathbf{n}}_\phi + B_{ca}\mathbf{n}_{wall}. \tag{5.9}$$

Combining Eq. (5.7) to (5.9) gives the coefficients A_{ca} and B_{ca} as

$$A_{ca} = \frac{\cos\theta - \cos\theta_0 \cos(\theta_0 - \theta)}{1 - \cos^2\theta_0}, \tag{5.10}$$

$$B_{ca} = \frac{\cos(\theta_0 - \theta) - \cos\theta_0 \cos\theta}{1 - \cos^2\theta_0}. \tag{5.11}$$

After A_{ca} and B_{ca} are found for each face on the wall boundary, the target normal vector $\mathbf{n}_\phi$ can be found and replaces the previous normal vector $\hat{\mathbf{n}}_{\phi,0}$. Recall that the curvature is calculated as $\mathcal{K} = -\nabla \cdot \mathbf{n}$ i.e the divergence of the interface normal vector. Any difference between $\mathbf{n}_\phi$ and $\hat{\mathbf{n}}_{\phi,0}$ leads to a local surface tension force, which then adjusts the local interface shape until the correct contact angle θ is enforced [115].

5.2.1 Yokoi dynamic contact angle formulation

In this work, the dynamic contact angle model proposed by Yokoi et al. [7] is implemented into the OpenFOAM solver. The dynamic contact angle θ_d is found as follows;

$$\theta_d = \begin{cases} min\left(\theta_e + (\frac{\mu u_{cl}}{\sigma k_a})^{\frac{1}{3}}, \theta_a\right) & \text{if } u_{cl} \geq 0 \\ max\left(\theta_e + (\frac{\mu u_{cl}}{\sigma k_r})^{\frac{1}{3}}, \theta_r\right) & \text{if } u_{cl} < 0, \end{cases} \tag{5.12}$$

where θ_r is the receding contact angle (measured experimentally during the recoil stage of the droplet impact), θ_a is the advancing contact angle (measured during the spreading stage of the droplet impact), θ_e is the equilibrium contact angle, u_{cl} is the contact line velocity and k_a and k_r are material related parameters to be adjusted to fit experimental results.

This model is based on Tanner's law [116]

$$Ca = k_m(\theta_d - \theta_e)^3, \tag{5.13}$$

for low Ca numbers where Ca is the capillary number $Ca = \mu u_{cl}/\sigma$ and k_m is an empirically-found material constant.

5.3 Validation

The CLSVOF implementation is validated using a number of tests, ranging from simple surface tension benchmarks such as the static droplet test to the more difficult droplet splashing test. In some sections in this chapter, two different CLSVOF solutions are shown, with CLSVOF-p using the LS field to calculate the curvature $\mathcal{K}(\phi)$ for the surface tension force, and CLSVOF-a using the VOF field to calculate $\mathcal{K}(\alpha)$. This allows us to compare the performance of the same interface reconstruction scheme, but with different approaches to the calculation of curvature. The curvature for interFoam and interFlow are calculated using $\mathcal{K}(\alpha)$.

5.3.1 Static droplet at equilibrium

A circular droplet at equilibrium is placed in a zero gravity field to assess the strength of spurious currents at the interface [67]. In this case, the momentum equation is reduced to

$$0 = -\nabla P + \mathbf{F}_\sigma, \tag{5.14}$$

where $\mathbf{F}_\sigma = \sigma\mathcal{K}\nabla\alpha$. The curl of Eq. (5.14) should have the curvature satisfied as

$$\nabla\mathcal{K} \times \nabla\alpha = 0. \tag{5.15}$$

As described in [117], spurious currents develop when this condition is not fulfilled. In this test case with a circular droplet, the surface should have a constant curvature and the flow is curl-free if the calculated curvature is constant. Therefore, this test compares the effect of curvature calculation methods on the generation of spurious currents between CLSVOF-p and CLSVOF-a, as well as with interFlow and interFoam.

The initial conditions are as follows; for the set-up as shown in Fig. 5.2. The validation is tested with a 50 x 50 mesh and 100 x 100 mesh, for $t = 0.01$. The analytical solution

Table 5.1: Material properties for the static droplet test

$\rho_l\ (kg/m^3)$	$\rho_g\ (kg/m^3)$	$\mu_l\ (\frac{kg}{ms})$	$\mu_g\ (\frac{kg}{ms})$	$g\ (m/s^2)$	$\sigma\ (N/m)$
1000	1	1×10^{-3}	1×10^{-5}	0	0.01

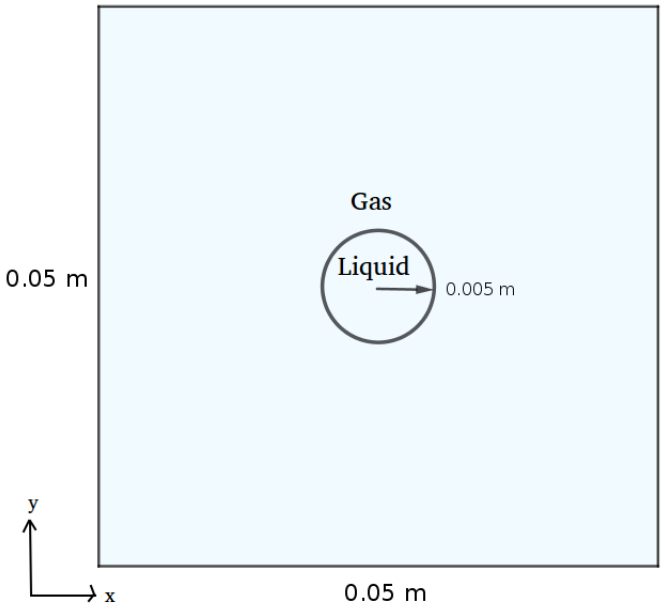


Figure 5.2: Numerical setup for the Laplace pressure test

for pressure jump can be obtained using the Laplace pressure theorem,

$$\Delta p = \sigma\left(\frac{1}{R}\right) \tag{5.16}$$

and the numerical pressure

$$\Delta p = p_0^{in} - p_\infty^{out} \tag{5.17}$$

where p_0^{in} is the pressure inside the droplet and p_∞^{out} is the pressure outside of the droplet. Eq. (5.16) gives $\Delta p = 2$ for this case. Fig. 5.3 show the pressure difference results taken across the x-axis at $y = 0.025$.

Results in Fig. 5.3(a) show that for the coarser mesh, CLSVOF-p, CLSVOF-a, and interFlow all produce a pressure differential closer to the exact solution than interFoam.

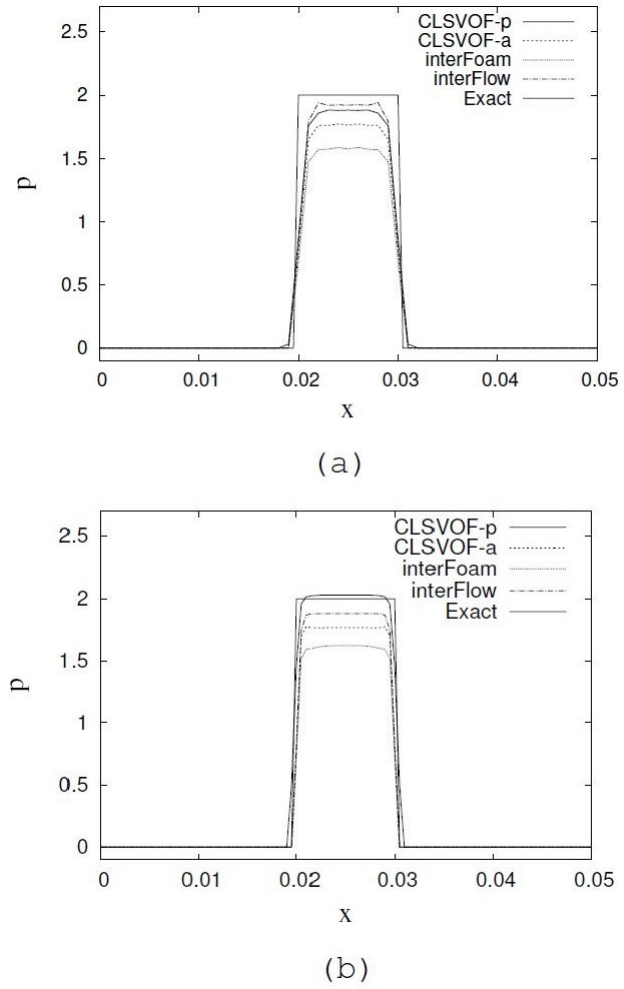


Figure 5.3: Pressure difference for the static droplet test for (a) 50x50 mesh, (b) 100x100 mesh.

On the finer mesh in Fig.5.3(b), it can be seen that CLSVOF-p produces results closest to the exact solution than the other three methods.

It can also be seen from Fig. 5.4 and Table 5.2 that CLSVOF-p has the smallest magnitude of spurious currents. This confirms that calculating curvature $\mathcal{K}$ using ϕ does indeed improve the results due to the smoother nature of the LS field. Refining the mesh to 100 x 100 has also shown to reduce the spurious currents, as in Fig. 5.5 and Table 5.3.

Table 5.2: Spurious currents in the static droplet test in 50 x 50 mesh, in $|U|(ms^{-1})$.

Method	min $\|U\|$	max $\|U\|$
CLSVOF-p	1.19×10^{-8}	8.12×10^{-4}
CLSVOF-a	9.61×10^{-8}	1.63×10^{-3}
interFlow	1.47×10^{-7}	1.17×10^{-3}
interFoam	4.92×10^{-7}	2.71×10^{-3}

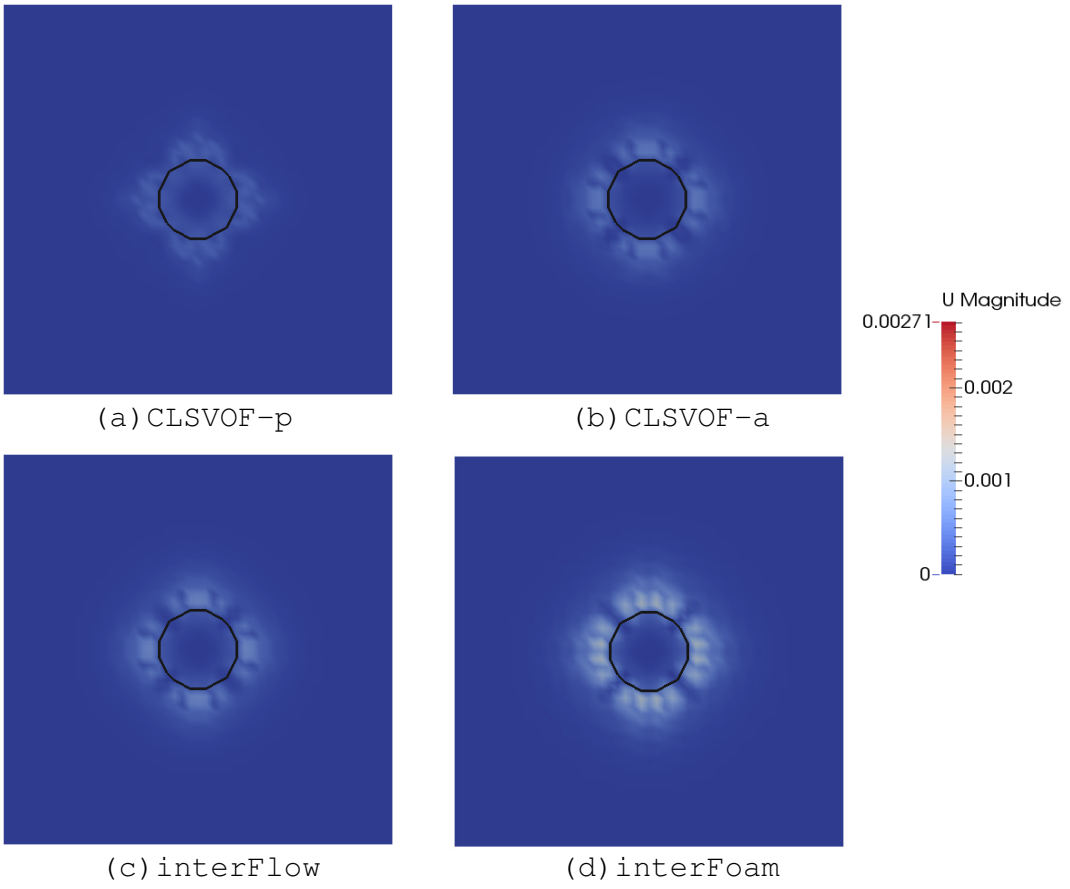


Figure 5.4: Spurious currents for static droplet test for 50 x 50 mesh

Table 5.3: Spurious currents in the static droplet test in 100 x 100 mesh, in $|U|(ms^{-1})$.

Method	min $\|U\|$	max $\|U\|$
CLSVOF-p	5.69×10^{-12}	6.19×10^{-4}
CLSVOF-a	5.99×10^{-12}	7.04×10^{-4}
interFlow	8.02×10^{-12}	8.32×10^{-4}
interFoam	9.27×10^{-12}	2.40×10^{-3}

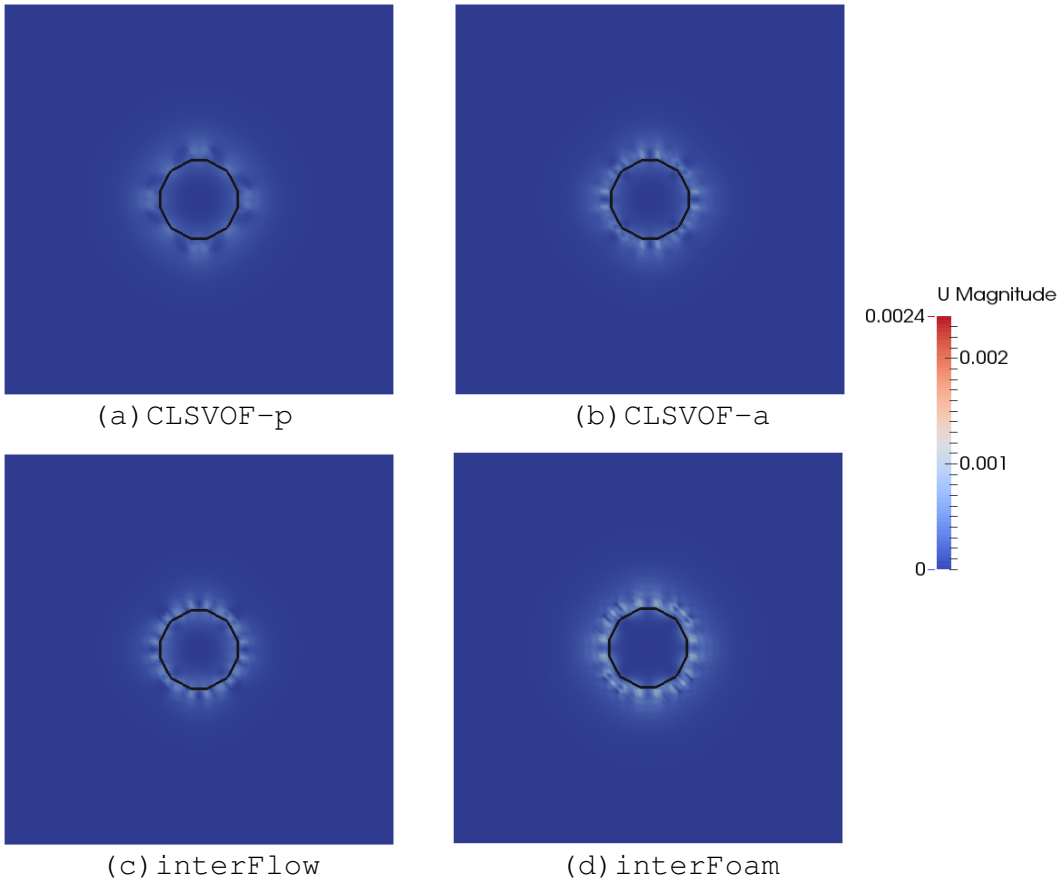


Figure 5.5: Spurious currents for static droplet test for 100 x 100 mesh

5.3.2 2D dam break

The scheme is tested using the classical dam break case in 2 dimensions, which is a common qualitative benchmark for free-surface flows. A static column of water under the influence of gravity is initialised on the left side of the tank. At time $t = 0$ of the simulation, the water column collapses, impacting an obstacle at the bottom of the tank. This case is considered to be sufficiently complicated, as it exhibits jet formation, breaking of waves, gas entrapment, and surface breakup. The set-up of the test case is shown in 5.6.

The simulations are compared to experimental data provided by Koshizuka and Oka [4], where a tank containing a column of water 0.25 m wide and 0.5 m high are held in place by a restraint, which is then removed to begin the simulation. The fluid properties are $\rho_L = 1000\ kg/m^3$, $\mu_L = 1.0 \times 10^{-3}\ kg/ms$, and $\sigma = 0.072\ kg/s^2$ for water in air and $\rho_G = 1\ kg/m^3$ and $\mu_G = 1.7 \times 10^{-5}\ kg/ms$ for air. Gravity is set to $9.8\ m/s^2$. No-slip boundary condition is applied to all walls, with the top of the tank being set as atmosphere.

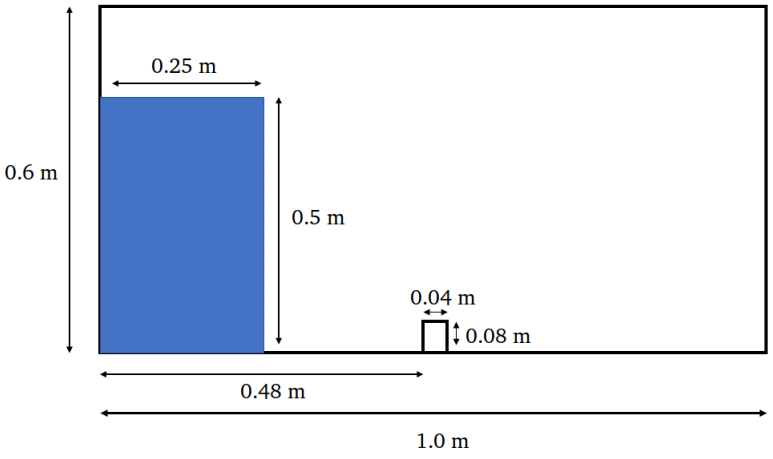


Figure 5.6: Schematic of the dam break set-up at $T = 0$ showing the liquid column on the left hand side.

As the interface deformation in this test case is more reliant on inertial forces rather than surface tension forces, results show that all methods produced good agreement with experimental findings. The finger-like projection is captured at $T = 4.043$ by all four schemes. While all four schemes result in the same profile, it is demonstrated that CLSVOF-p, CLSVOF-a, and interFlow can maintain the sharpness of the fluid interface, and interFoam shows diffusion in Figures (5.7(e),(j)). With this encouraging validation results, we may proceed to more difficult tests.

5.3.3 2D rising bubble

In 2009 Hysing et al [5] published a numerical benchmark for two test cases of a 2 dimensional rising bubble. The bubble is initially centred at $(x, y) = (0.5, 0.5)$ with an initial radius of $r = 0.25$.

Table 5.4: Material properties for 2D rising bubble test cases 1 and 2

Case	ρ_L (kg/m^3)	ρ_G (kg/m^3)	μ_L (kg/ms)	μ_G (kg/ms)	g (m/s^2)	σ (kg/s^2)	Eo
1	1000	1	10	0.1	-0.98	1.96	124.5
2	1000	100	10	1	-0.98	24.5	9

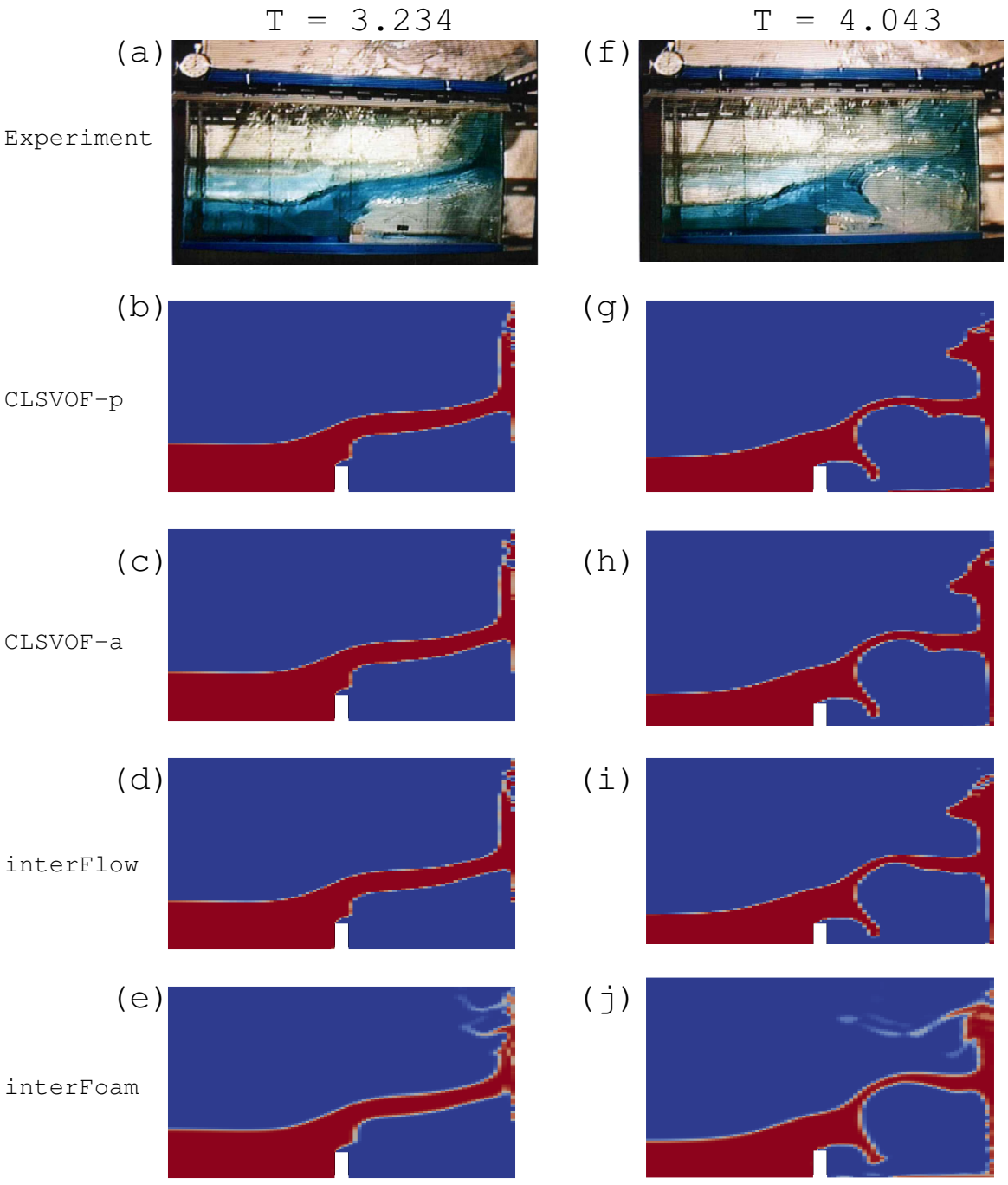


Figure 5.7: Dam-break simulation results at T=3.234 and T=4.043 compared to experimental results by [4].

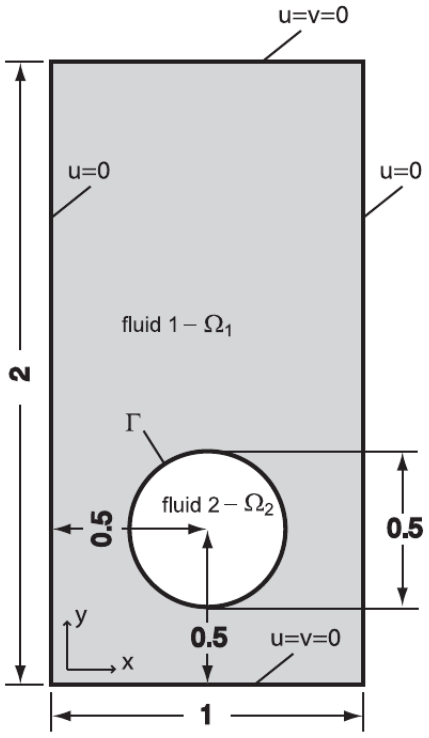


Figure 5.8: Numerical setup for the Hysing rising bubble test, figure reproduced from [5] .

The Eötvös (Eo) number is a dimensionless quantity given by

$$Eo = \frac{\Delta\rho g L^2}{\sigma} \tag{5.18}$$

where $\Delta\rho$ is the difference in density between the two phases (kg/m^3), g is gravitational acceleration (m/s^2), L is characteristic length (m), and σ is the surface tension coefficient (N/m). It measures the importance of gravitational forces compared to surface tension forces; a lower Eötvös value indicates that the case is surface-tension dominant. Therefore Case 2 should be more affected by surface tension forces than Case 1, meaning that the effect of the curvature calculation should be more pronounced in Case 2. Case 1 however, has a larger density ratio between the fluid and gas which is also difficult to simulate.

The results obtained in this work are compared against those presented by Hysing et al.

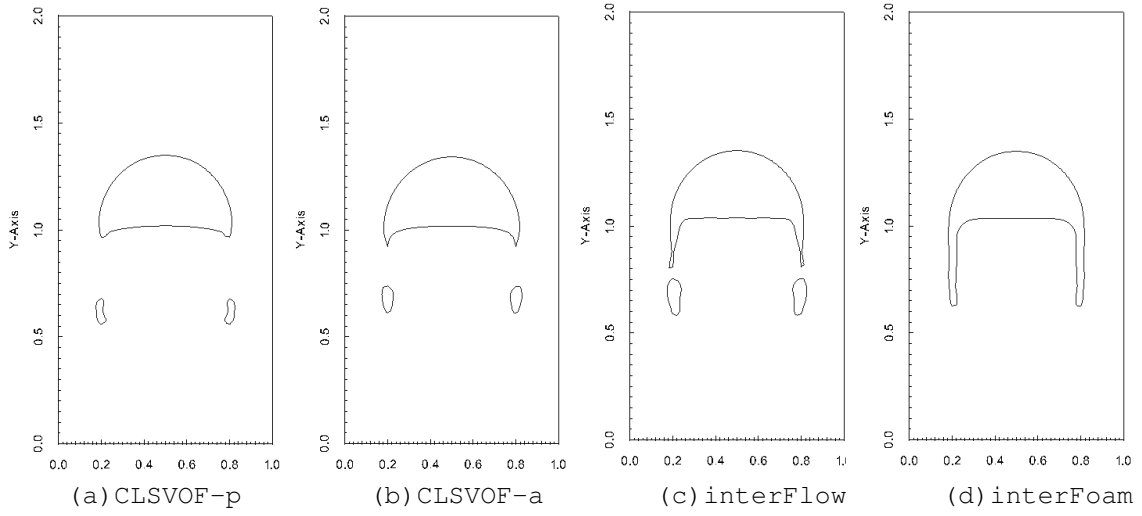


Figure 5.9: Case 1: Final position of bubble at T=3 depicted with contour $\alpha = 0.5$ using a 50 x 100 mesh.

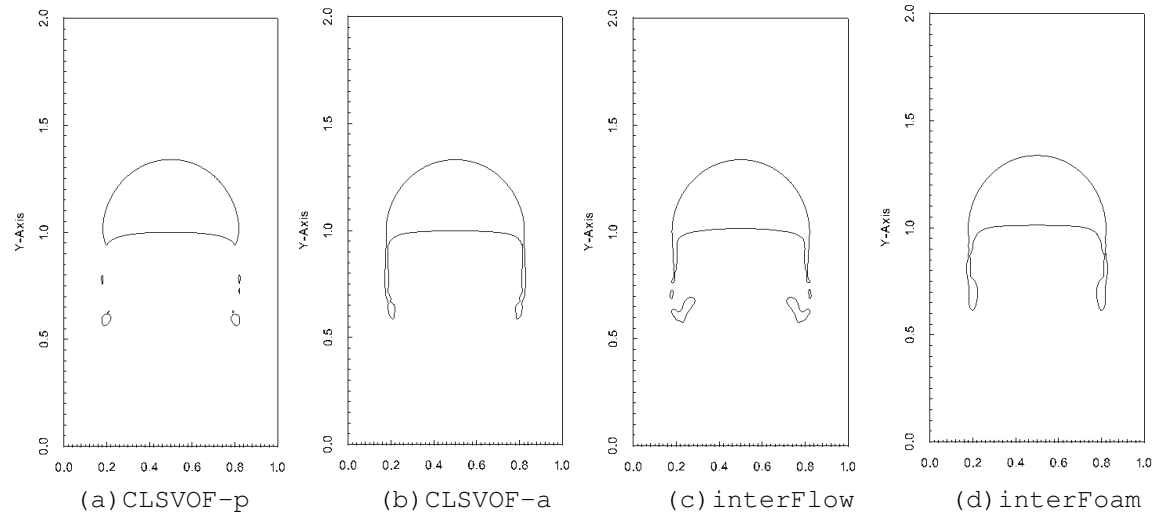


Figure 5.10: Case 1: Final position of bubble at T=3 depicted with contour $\alpha = 0.5$ using a 100 x 200 mesh.

[5]. The simulations are performed using two mesh sizes; 50×100 with $\Delta t = 0.0025$ s, and 100×200 with $\Delta t = 0.00125$ s.

Case 1

Figures (5.9) and (5.10) show the results of test Case 1 for 50x100 and 100x200 meshes

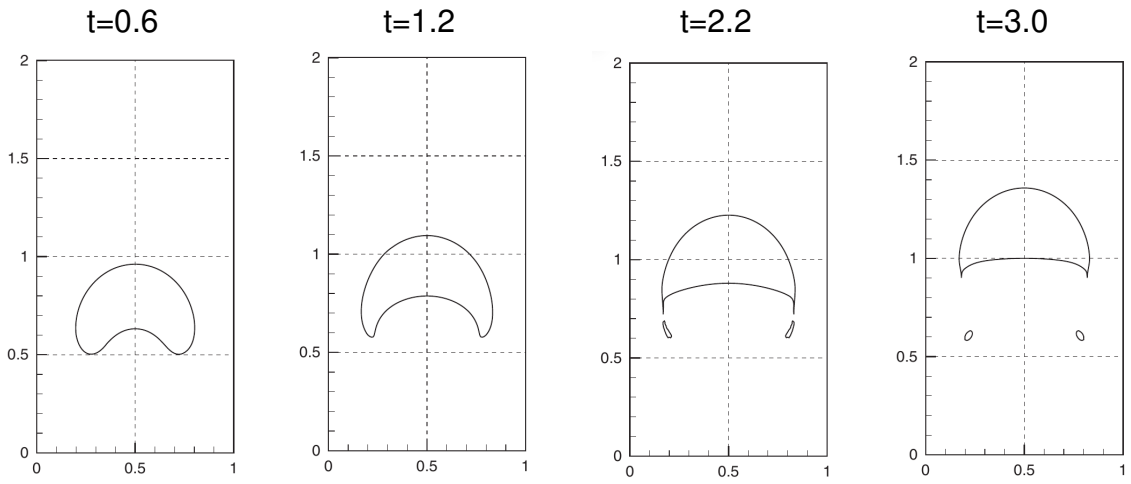


Figure 5.11: Benchmark results of the rising bubble test Case 1, reproduced from Hysing et al [5].

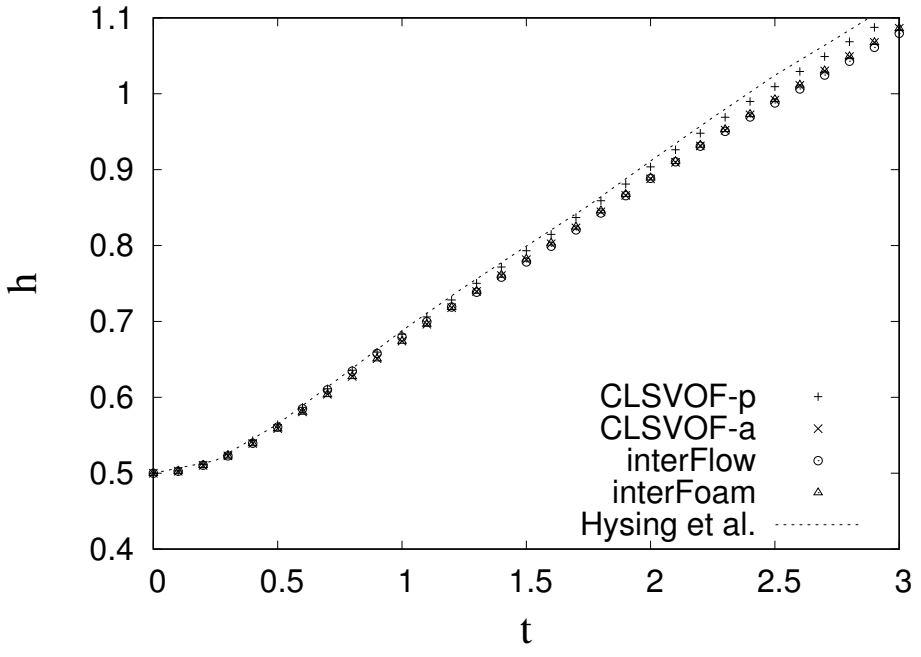


Figure 5.12: Case 1: Position of mass centre of bubble against time using a 50 x 100 mesh compared to a reference solution by Hysing et al. [5] .

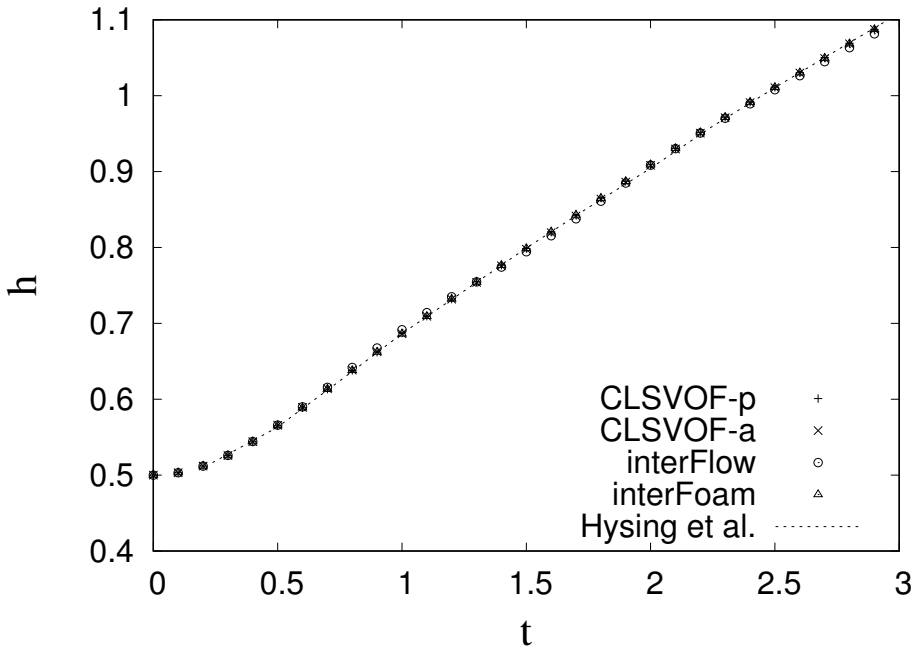


Figure 5.13: Case 1: Position of mass centre of bubble against time using a 100 x 200 mesh compared to a reference solution by Hysing et al. [5] .

respectively. This test case is less surface-tension dominant and all four schemes have managed to produce a solution. This test however, is more buoyancy-dominant due to the higher density ratio between the two fluids. The bubble will accelerate from its initial position, and this causes a greater pressure gradient on the lower surface of the bubble compared to its top surface. A vortex sheet develops and causes a jet of water to push the bubble from its lower surface. The circulation current under the bubble causes the bubble skirt to fold in, and eventually detaches, as seen in Hysing et al. [5]

In Fig. 5.11, over time it can be seen that the initially circular bubble evolved into a 'jellyfish' shape, with a bubble skirt forming around T=2.2. The solution by CLSVOF-p results in detached bubbles resulting from the bubble skirt pinching off. CLSVOF-a produces detached bubbles in the coarser mesh, but the bubble skirt remained intact in the finer mesh. interFlow produces solution with detached bubbles but with the bubble skirt still discernible. Finally, interFoam produces no detached bubbles in both mesh resolutions.

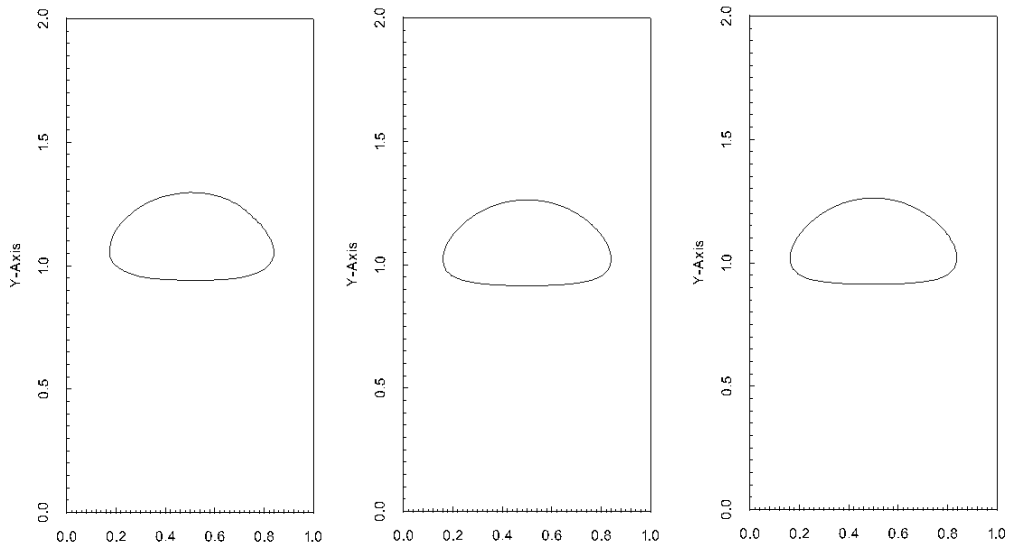


Figure 5.14: Case 2: Final position of the bubble at $t = 3$ depicted with contour $\alpha = 0.5$ using a 50 x 100 mesh.

Upon mesh refinement, the shape changes for all solutions. While CLSVOF-p shows the closest agreement in shape to the Hysing reference, Hysing also mentioned that there is no agreement on the final shape of the bubble even among the schemes tested in their work.

The position of the mass centre of the bubble (Fig. 5.12) is tracked for Case 1 and CLSVOF-p is found to produce the closest agreement to Hysing, with interFlow being the least in agreement. Refining the mesh to 100x200 (Fig. 5.13) is shown to improve agreement for all four schemes, again with the CLSVOF-p result showing the closest agreement with the reference solution.

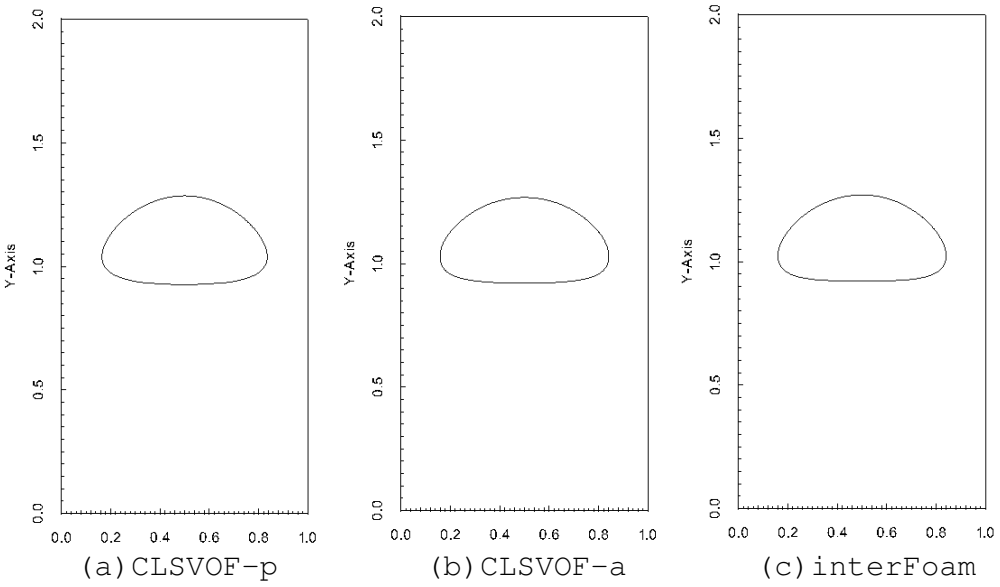


Figure 5.15: Case 2: Final position of the bubble at $t = 3$ depicted with contour $\alpha = 0.5$ using a 100 x 200 mesh .

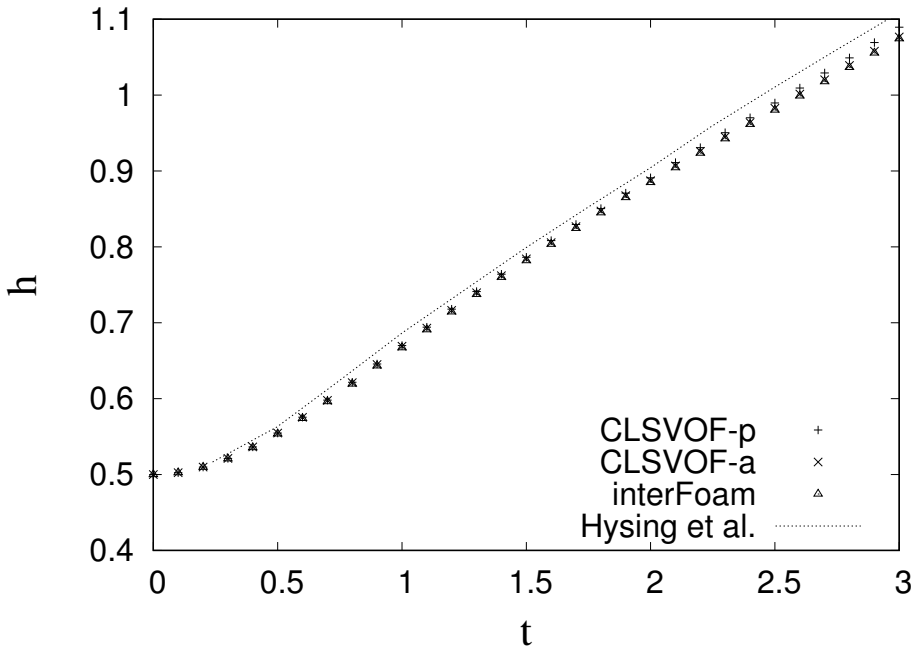


Figure 5.16: Case 2: Position of mass centre of bubble against time using a 50 x 100 mesh compared with reference solution by Hysing et al. [5] .

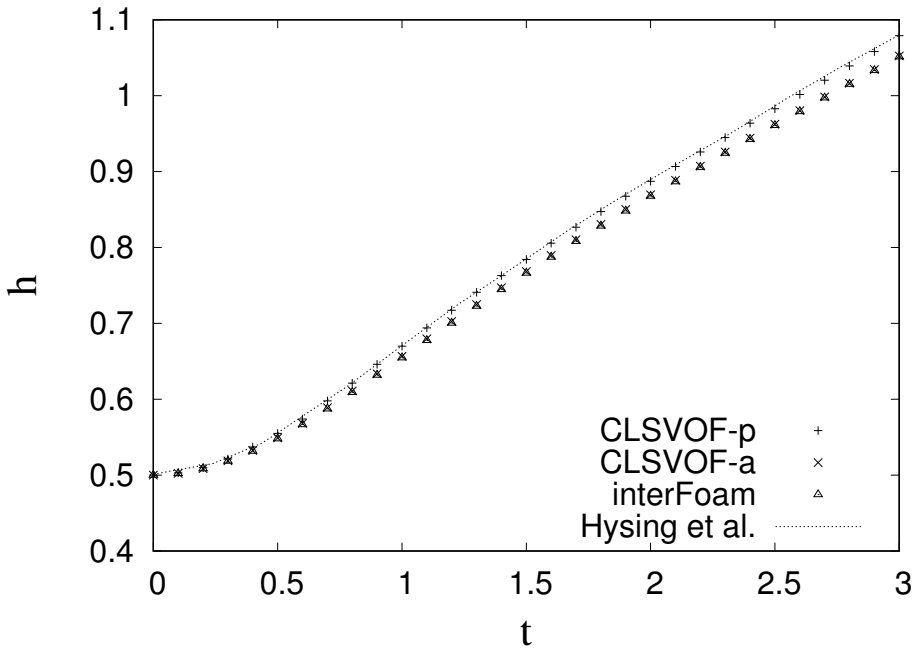


Figure 5.17: Case 2: Position of mass centre of bubble against time using a 100 x 200 mesh compared with reference solution by Hysing et al. [5] .

Case 2

As can be seen in the Fig. 5.14, CLSVOF-a, CLSVOF-p, and interFoam managed to achieve a similar final bubble shape at T=3 using a 50x100 mesh for Case 2 which is more surface tension dominant. The final form of the bubble does not deform as much as Case 1 as the surface tension force dominates over buoyancy in this test. In this test case, interFlow has failed to produce a solution. Seeing as interFlow is a geometric VOF method, the interface may be locally discontinuous, which may potentially cause an inaccurate estimate of surface tension force.

The mass centre of the bubble as it moves upwards in the y-direction is tracked for all simulations and is plotted as in Fig. 5.16. The reference solution is taken from Hysing et al. As can be seen, CLSVOF-p agrees most closely to the reference solution, followed by CLSVOF-a and finally interFoam.

Upon mesh refinement to 100x200, there is still no major difference in the final bubble

shape across all schemes (Fig. 5.15). However, the position of the mass centre as plotted in Fig. 5.17 shows that CLSVOF-p agrees very well with the reference solution with considerable improvement compared to the coarser mesh.

Overall, the two test cases in this test have shown that CLSVOF-p shows the best potential out of all four methods compared in simulating surface-tension dominant flows as well as high density-ratio flows.

5.3.4 Rayleigh-Taylor instability

In 1878, Rayleigh [118] investigated a hydrostatic instability phenomenon that occurs when two fluids mix together when the heavier of the two fluids are placed above the lighter one. The perturbation that develops along the interface is now referred to as the Rayleigh-Taylor instability. The problem is set with an initial perturbation,

$$y(x) = 2 + 0.1\cos(2\pi x/d) \tag{5.19}$$

where the instability is expected to progress into a mushroom-like structure as the heavier fluid on the top part moves downwards due to gravity. The domain is given as [0,1]x[0,4] (or $[0, d] \times [0, 4d]$ with a mesh of 112×448. The densities are $\rho_1 = 3$ and $\rho_2 = 1$ giving a density ratio of 3. The viscosities are $\mu_1 = \mu_2 = 0.0031316$, gravity $g = -9.81$, and surface tension coefficient is set to $\sigma = 0.01$. The Reynolds number in this case is defined as

$$Re = \frac{\rho_1 d^{3/2} g^{1/2}}{\mu_1} = 3000, \tag{5.20}$$

and the Atwood number, which is a dimensionless number describing the density ratio of fluids, is defined as

$$\mathcal{A} = \frac{\rho_1 - \rho_2}{\rho_1 + \rho_2} = 0.5. \tag{5.21}$$

In Rayleigh-Taylor instability, the penetration distance of the heavy fluid bubbles into the light fluid can be expressed as a function of $\mathcal{A}gt^2$ [119]. For $\mathcal{A}$ closer to 0, the resulting flows tend to form symmetric 'fingers' of fluid, whereas for $\mathcal{A}$ closer to 1, the lighter fluid (usually placed 'below' the heavier fluid) forms larger plumes [120].

In order to validate the results, we compare against the works of Tryggvason [121] and Guermond [122] as previously done by Hosseini [123].

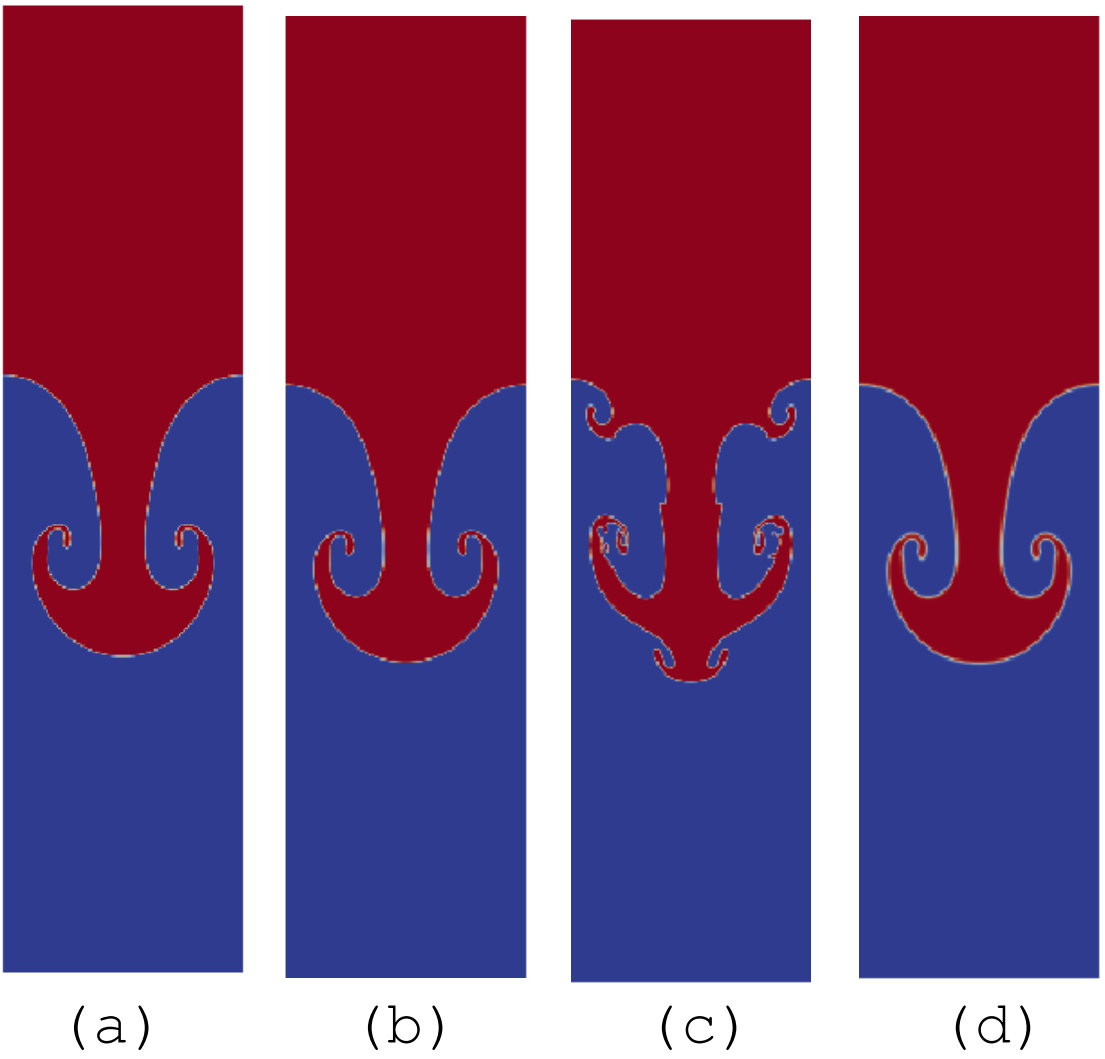


Figure 5.18: Comparison at t = 1.66 between (a) CLSVOF-p, (b) CLSVOF-a, (c) interFlow, and (d) interFoam .

Fig. 5.18 shows the progress of the instability at t=1.66 where a mushroom-like structure begins to develop. CLSVOF-p, CLSVOF-a, and interFoam show similar profiles, with the interFoam solution being more diffusive across the fluid interface. The solution obtained by interFlow shows extra finger-like projections which is not observed in the other solutions, nor in the reference solution provided by [122].

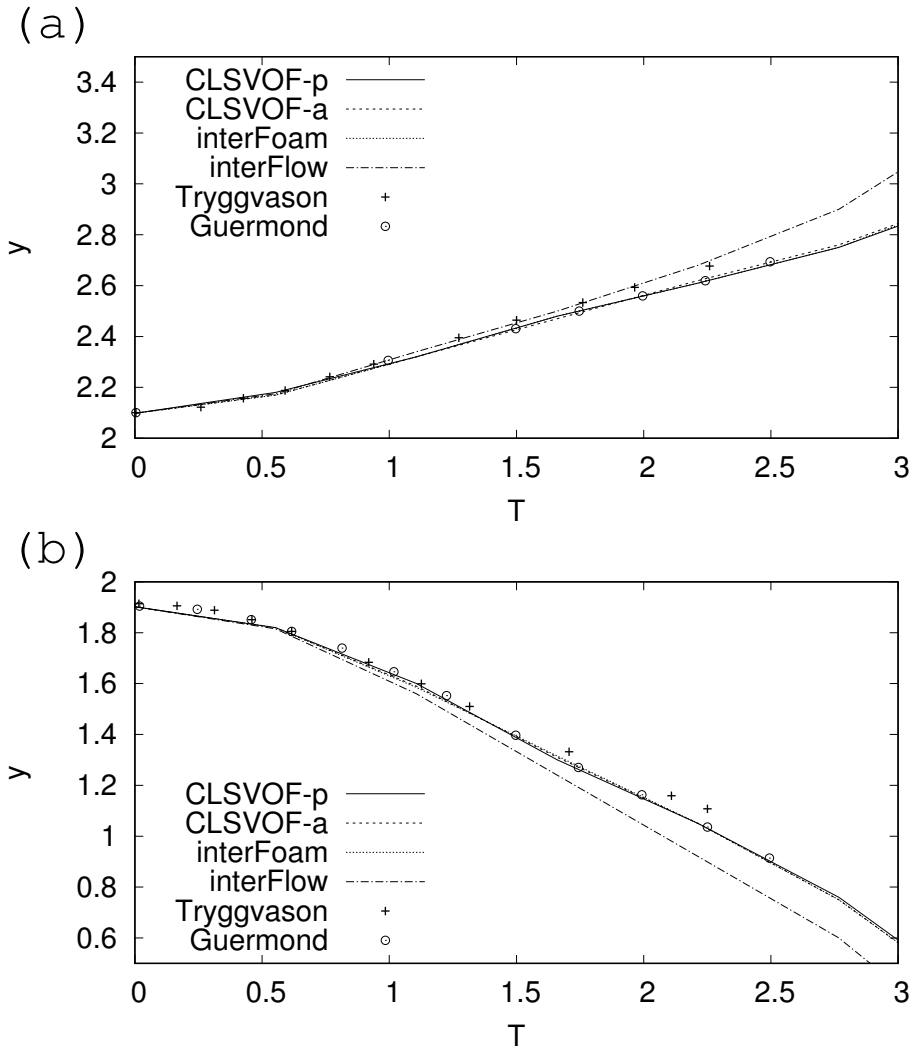


Figure 5.19: The y-coordinate of the tip of the (a) rising and (b) falling fluid against time.

The y-coordinate of the tip of the rising and falling fluid is plotted against time in Fig. 5.19 to quantitatively compare with Tryggvason and Guermond. Note that the reference provided by Tryggvason neglected the effects of surface tension force while Guermond had the surface tension force accounted for. The solution obtained by CLSVOF-p, CLSVOF-a, and interFoam agree well with Guermond's reference res-

ults for the rising tip of the fluid, while the interFlow solution deviates slightly from it and is closer to Tryggvason's reference.

Fig. 5.19(b) shows the comparison of the tip of the falling fluid. Again, it is shown that CLSVOF-p, CLSVOF-a, and interFoam agree well with Guermond's reference. The result obtained by interFlow again shows deviation away from both Guermond and Tryggvason.

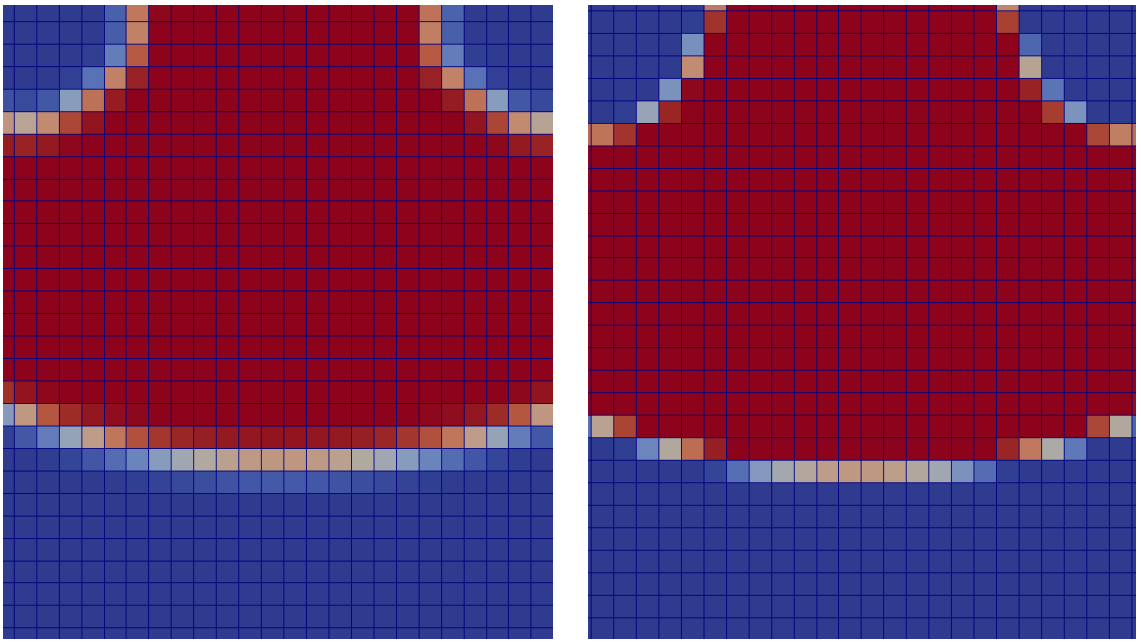


Figure 5.20: Zoomed in view of the Rayleigh-Taylor instability at $t = 3.32\tau$ obtained using (a) interFoam and (b) CLSVOF-p.

Fig. 5.20 demonstrates the sharpness of the fluid interface obtained by CLSVOF-p, compared to the more diffusive solution by interFoam. The fluid interface obtained by interFoam can be seen spreading over two mesh cells.

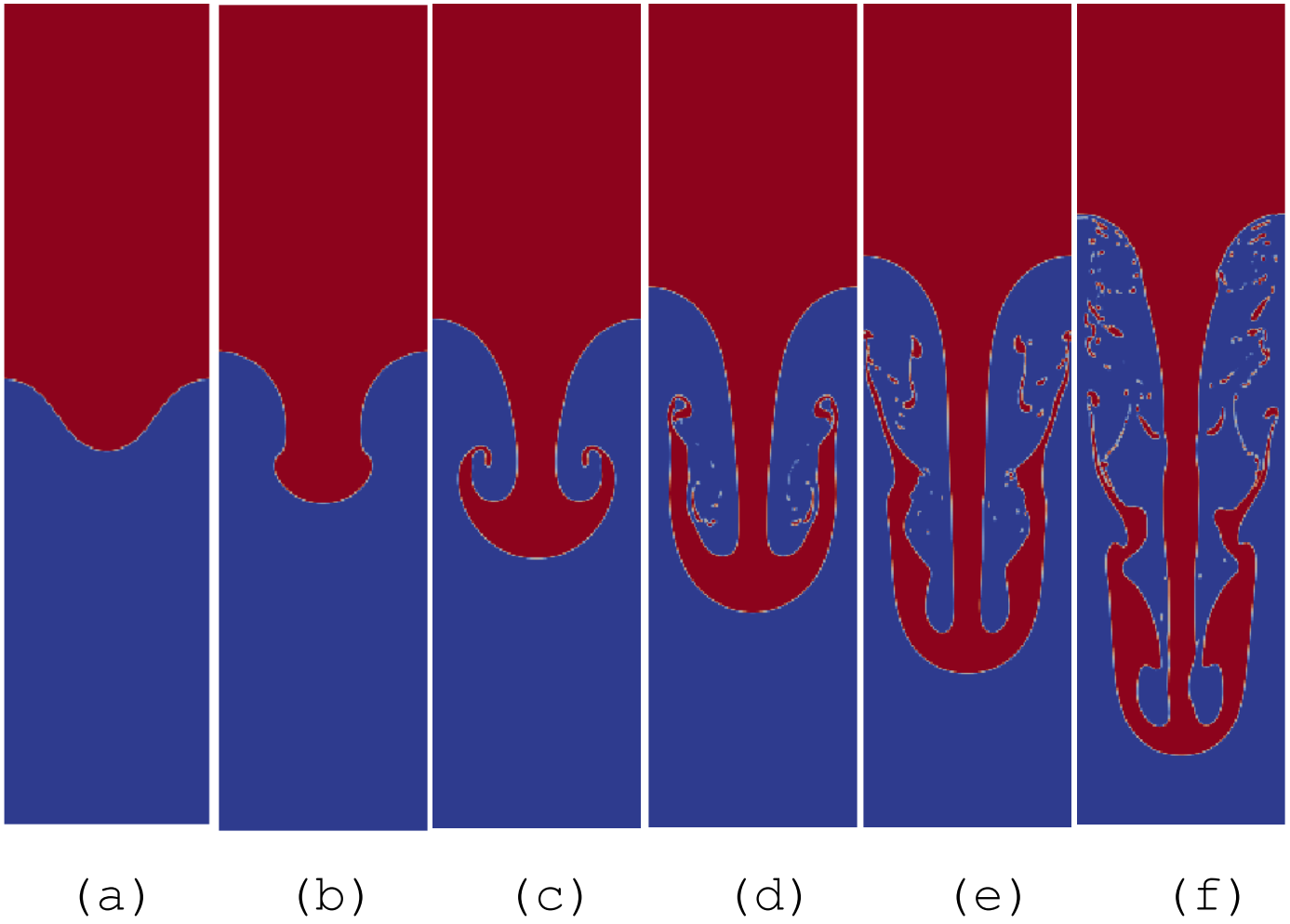


Figure 5.21: The evolution of a single-wavelength initial condition in the Rayleigh-Taylor instability test using CLSVOF-p, with mesh 112 x 448 at times (a) 0.55τ**, (b)** 1.10τ**, (c)** 1.66τ**, (d)** 2.21τ**, (e)** 2.76τ**, (f)** 3.32τ **.**

5.3.5 Droplet splash on dry surface

The proposed method is validated for a droplet impact phenomenon. Understanding the phenomena of impacting droplets is vital for many industrial processes such as painting and the cooling of hot surfaces [124]. In these processes, the liquid is required to spread in an even manner, to cover as large an area as possible. While high impact velocities promote droplet spreading and greater area coverage, they also cause splashing which is undesirable as the satellite droplets that form during splashing bounce off and do not ultimately deposit on the surface.

This phenomenon is a complex one as the resulting splash behaviour is a result of a complex interaction among physical effects such as inertia, viscosity, surface tension, gravity, contact angle, and surface roughness [125]. Therefore, the ability of a fluid solver to accurately simulate phenomena such as droplet splashing would mean it has great potential to be used for complex multiphase flow problems.

In this test, the proposed method is validated against an experiment by Tsai et al. [6] where a distilled water droplet impacts a super-hydrophobic substrate. The fluid properties are $\rho_{liquid} = 1000kg/m^3$, $\rho_{air} = 1.25kg/m^3$, $\mu_{liquid} = 1.0 \times 10^{-3}Pa \cdot s$, $\mu_{air} = 1.82 \times 10^{-2}Pa \cdot s$, $\sigma = 7.2 \times 10^{-2}N/m$, $g = 9.8m/s^2$. The initial droplet diameter is $D = 1.86$ mm with impact speed 2.98 m/s. An constant contact angle of 163° is used since the dynamic contact angle was not measured in the experiment. The mesh used is $D = 75h$.

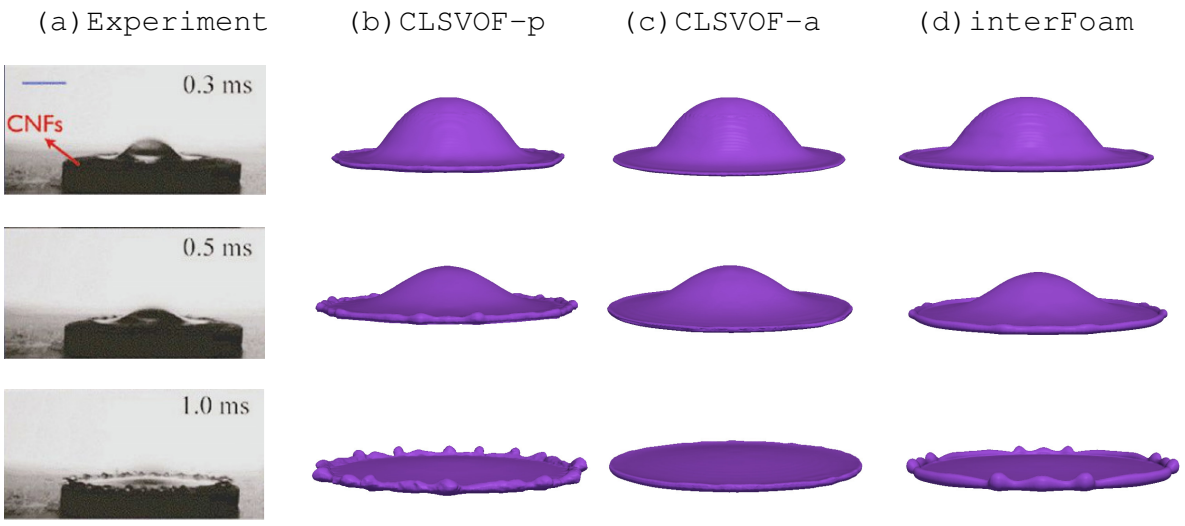


Figure 5.22: Comparison between (a) Experiment from Tsai et al [6] (b) CLSVOF-p (c) CLSVOF-a (d) interFoam for a droplet impacting a dry surface with static contact angle 163°.

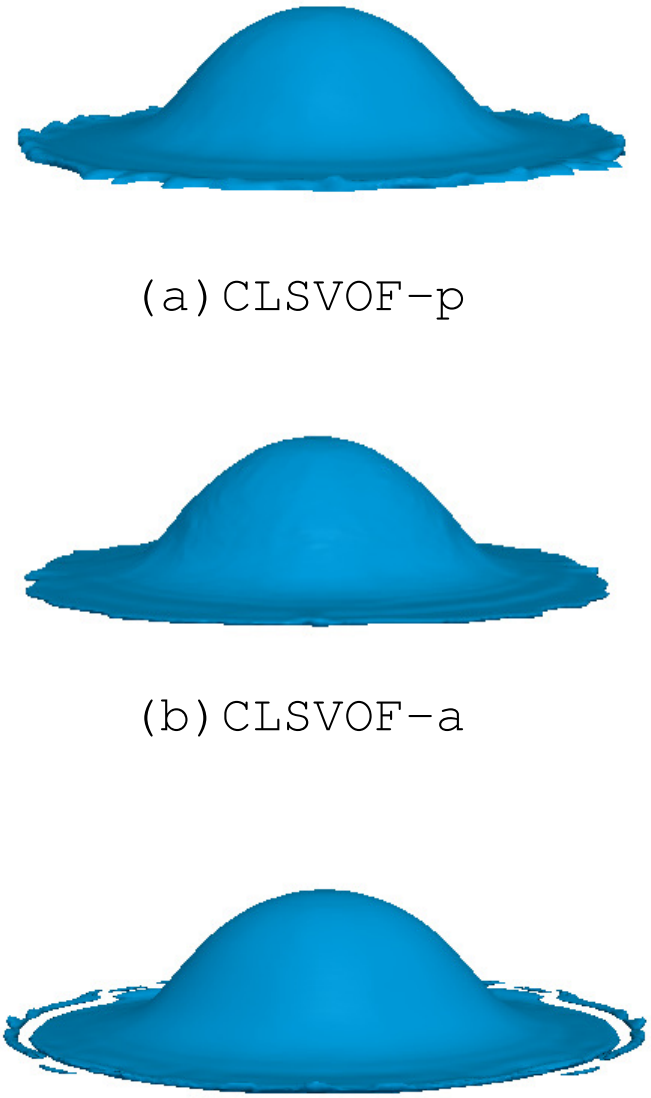
Fig. 5.22 shows the results of the water droplet impact for CLSVOF-p, CLSVOF-a, and interFoam compared against the experimental findings of [6]. The interFlow solver did not manage to produce a physical solution for this test. The simulation produced by the proposed CLSVOF-p solver produces good qualitative agreement with the experimental findings. Photographs from the experiment shows that on impact, a thin axisymmetric sheet of the droplet fluid spreads radially outward over the surface as a reaction to a sudden increase in pressure. Azimuthal undulations emerge around the leading edge of the radial spreading of the droplet. The perturbations grow larger and form finger-like structures protruding from the spreading droplet. CLSVOF-p has captured the perturbed leading edge (seen very clearly at $T = 1.0$ ms in Fig. 5.22) and agrees closely with the experiment snapshots.

interFoam and CLSVOF-a, both of which use the VOF field to calculate the local mean curvature, has failed to capture the undulations though interFoam performs slightly better than CLSVOF-a in this respect. These results show that calculation of mean curvature $\mathcal{K}$ plays a big role in ensuring the accuracy of simulations; using $\mathcal{K}(\phi)$ as

opposed to $\mathcal{K}(\alpha)$ have improved the quality of the simulation significantly when compared to the experiment. They also demonstrate that the proposed method is capable of simulating droplet impact phenomena accurately at least at a qualitative level.

A similar simulation is run using the same parameters as the previously described droplet test, but using the Yokoi dynamic contact angle model implemented in this work. The advancing contact angle θ_A is set to 160°, and the receding contact angle θ_R is 30°, with an equilibrium contact angle θ_e of 40°.

Results in Fig. 5.23 again show that only CLSVOF-p managed to capture the formation of the finger structures at the leading edge of the droplet. Results obtained by CLSVOF-a show the beginning of finger-like structures but not as clearly resolved as in CLSVOF-p. Results obtained by interFoam show a halo-like structure around the leading edge of the droplet, with no finger structures on the main body of the droplet. With the results obtained so far, CLSVOF-p shows the greatest promise among the three schemes tested for simulating droplet splashing problems.



(a) CLSVOF-p

(b) CLSVOF-a

(c) interFoam

Figure 5.23: Droplet splashing using a dynamic contact angle model at $T = 0.0004$ s.

5.3.6 Droplet impact on a hydrophobic surface

To further validate the implementation of the dynamic contact angle model, the proposed method is compared against the experiment by Yokoi et al. [7], which tested the behaviour of a droplet impacting a dry surface. A spherical water droplet of $D = 2.28mm$ is impacted at 1 m/s onto a silicon wafer grafted with hydrophobic silane, giving a very smooth surface. The fluid properties are $\rho_L = 1000\ kg/m^3$, $\mu_L = 0.001\ Pa.s$, $\sigma = 0.072\ N/m$ for water, and $\rho_G = 1.25\ kg/m^3$, $\mu_G = 1.82 \times 10^{-5}\ Pa.s$. The mesh sized used is $80 \times 64 \times 80$. The contact angle parameters are set as $\theta_a = 114$, $\theta_r = 52$, and $\theta_s = 90$, with the constants $k_a = 9 \times 10^{-9}$ and $k_r = 9 \times 10^{-8}$.

Using these test parameters, the droplet does not exhibit splashing behaviours as in Section 5.3.5; instead, at the given low impact velocity, the droplet firstly spreads radially across the surface. As seen in Fig. 5.25, the maximum contact diameter (the diameter of the cross-section of the droplet that is in contact with the surface) is achieved at around $T = 4$ ms. This is captured qualitatively and quantitatively by CLSVOF-p, CLSVOF-a, and interFoam; interFlow did not succeed in producing a solution for this test. Afterwards the droplet starts contracting, and this is where the discrepancy between each simulation result and the experiment begins. The experimental finding shows that the minimum contact diameter of about 2 mm is achieved at around $T = 2$ ms. However, the simulation results obtained by CLSVOF-p and CLSVOF-a show an overestimation of the minimum contact diameter (with both showing a minimum of around 2.4 mm). The simulation obtained using the interFoam scheme has greatly underestimated the rate of contraction of the droplet as can be seen between $T = 1$ ms and $T = 1.5$ ms in Fig. 5.25. The minimum contact diameter is also not achieved by interFoam. At $T = 1.8$ ms, the droplet is expected to experience a second spreading period which should see the contact diameter achieve a second maximum of 3.2 mm at $T = 2.2$ ms. This second spreading is not captured well by any of the schemes, but CLSVOF-p agrees most closely to experiment. Afterwards the droplet is shown to head towards a steady state where the contact diameter starts to level-off at 3 mm.

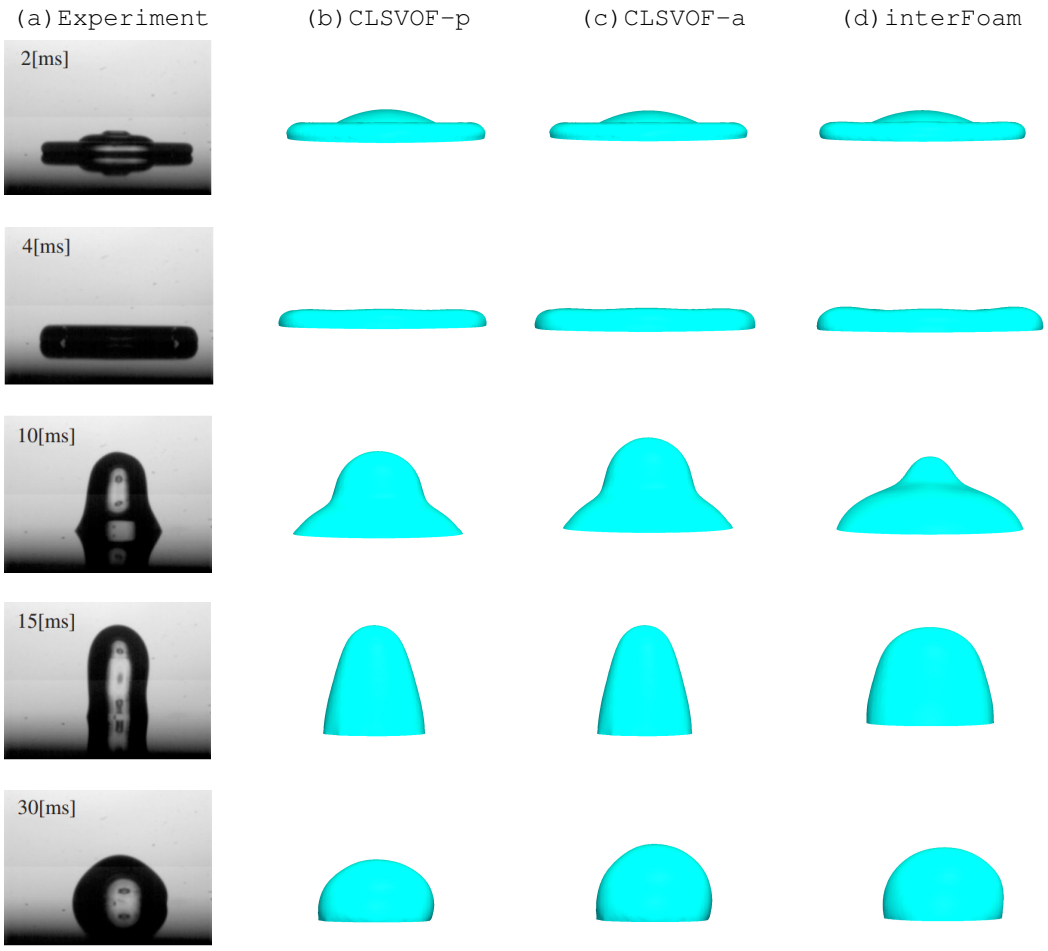


Figure 5.24: (a) Experiment from Yokoi et al [7] and simulations using (b) CLSVOF-p (c) CLSVOF-a (d) interFoam with an $80 \times 64 \times 80$ grid.

Again, CLSVOF-p shows the most reasonable agreement to experiment out of the three schemes tested.

5.3.7 Droplet splashing on fluid

The proposed method is validated against a droplet splashing phenomenon. The most interesting characteristic of a splashing droplet is the crown formation, which has been investigated by many researchers [8] [126] [127]. The instability mechanism that un-

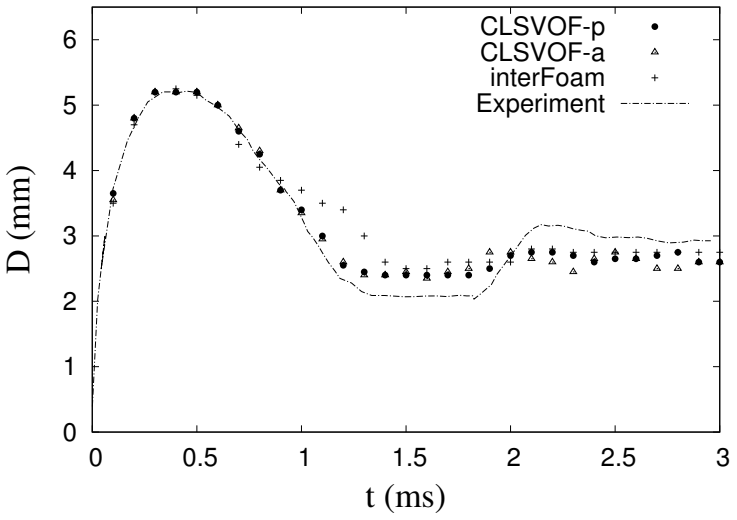


Figure 5.25: Comparison of droplet diameter against experimental data [7]

derlies the formation of the crown is still an active topic of study among researchers in the recent years [128] [129].

We validate the proposed method against the classical single droplet impingement test, where a droplet is made to impact a still liquid film. This test is expected to demonstrate the capability of a simulation to capture the structures of a liquid crown formation i.e. jets and droplets (Fig. 5.26 (a)). A droplet of diameter $D = 3.82$ mm with speed U = 3.56 ms^{-1} is made to impact a still liquid film of the same fluid of thickness 2.3 mm in a container of 0.01 m $\times 0.01$ m $\times 0.01$ m with mesh $80 \times 80 \times 80$. The fluid properties in this test are $\rho_L = 1000\ kg/m^3$, $\mu_L = 0.001\ Pa.s$, $\sigma = 0.072\ N/m$ for water, and $\rho_G = 1.25\ kg/m^3$, $\mu_G = 1.82 \times 10^{-5}\ Pa.s$. This setting corresponds to We = 670 and is run to T= 0.01 s.

The same setting is used to validate the interFoam solver by Deshpande et al. [63], which shows qualitative agreement for the interFoam solver results.

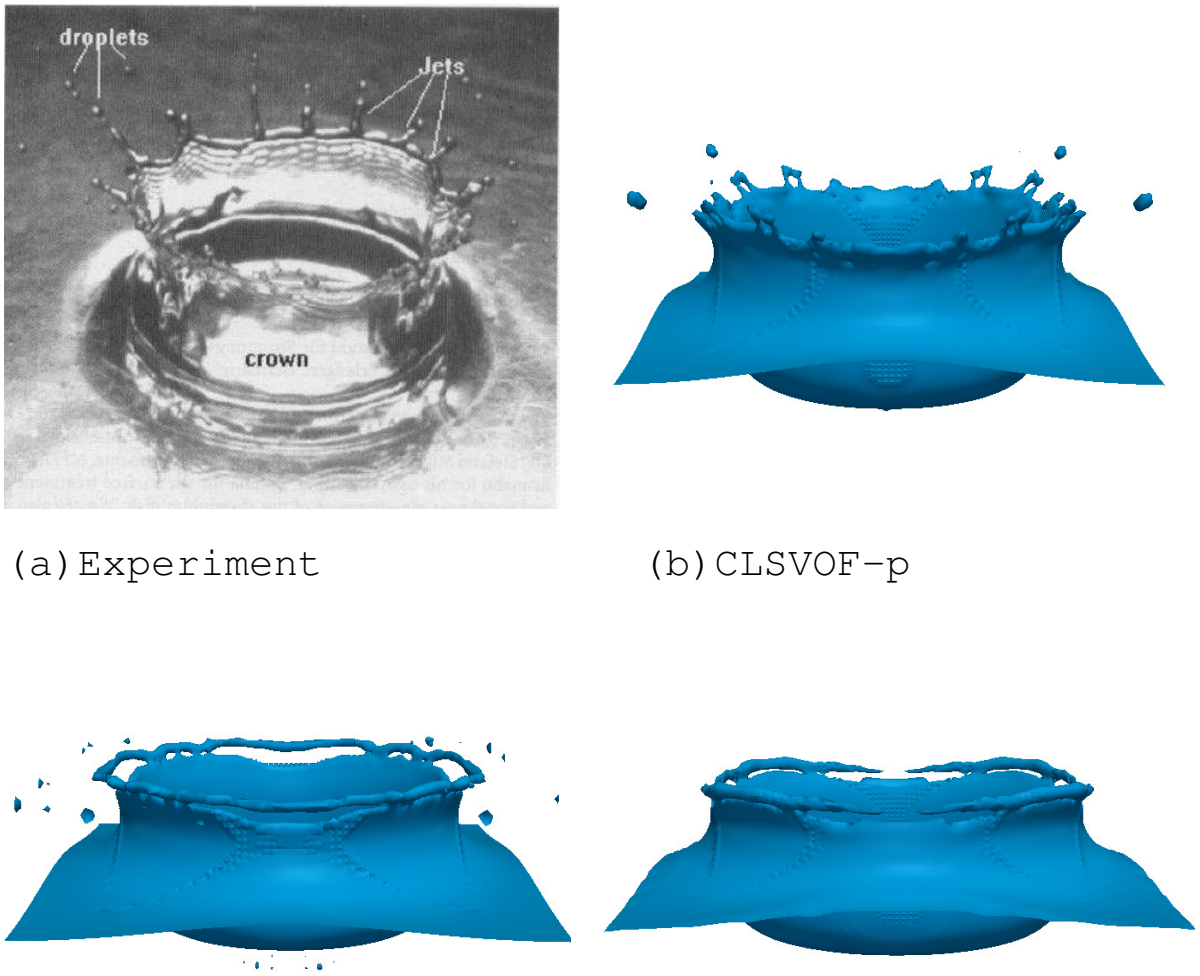


Figure 5.26: (a) Experiment from Cossali et al. [8] (b) CLSVOF-p (c) CLSVOF-a (d) interFoam.

The results obtained using CLSVOF-p shows the best qualitative agreement with the experiment results. The liquid crown is clearly formed, with the jet structures clearly resolved. The pinching phenomenon that leads to the formation of secondary droplets are also captured by CLSVOF-p alone. Neither CLSVOF-a nor interFoam managed to capture the crown formation in this test, and interFlow did not manage to produce a meaningful solution.

5.3.8 Binary droplet collision

Droplet collisions are very commonly observed in nature and have held the interest of researchers for a long time. The earliest investigation of droplet collisions date back to Rayleigh's [130] observation that droplets of rain bounce upon impact on a pool of water instead of going straight in. Gunn in 1965 [131] investigated the characteristics of freely falling water droplets and found that the collisions of two water droplets (binary collisions) can result in four outcomes, which are 'bouncing', 'coalescence', 'disruption after coalescence', and 'drop spatter'.

Droplet collision can be considered as a part of the process of liquid atomisation, which is a field of huge interest in engineering. As the bulk liquid is atomised into droplets, the way these droplets collide with each other will affect the final droplet size in the spray.

In 1989, Ashgriz and Poo [9] produced extensive experimental data for the binary collision of water drops, of Weber numbers ranging between 1 to 100. Their publication identified one coalescence regime and two different droplet separation regimes occurring at around the same range of We numbers, but at different impact parameters. These regimes are the reflexive separation regime and the stretching separation regime (Fig. 5.27). The former is expected to occur for near head-on collisions, and the latter for larger offset impacts.

In this section, the proposed method is tested and compared against the experimental results from [9] for the separation regimes. The material properties used are as follows;

Table 5.5: Material properties for the droplet collision tests

ρ_L (kg/m^3)	ρ_G (kg/m^3)	μ_L (kg/ms)	μ_G (kg/ms)	g (m/s^2)	σ (kg/s^2)
1000	1.25	0.001	1×10^{-6}	-9.81	0.072

where Fluid L is the test fluid (water), and Fluid G is the surrounding gas. The test is

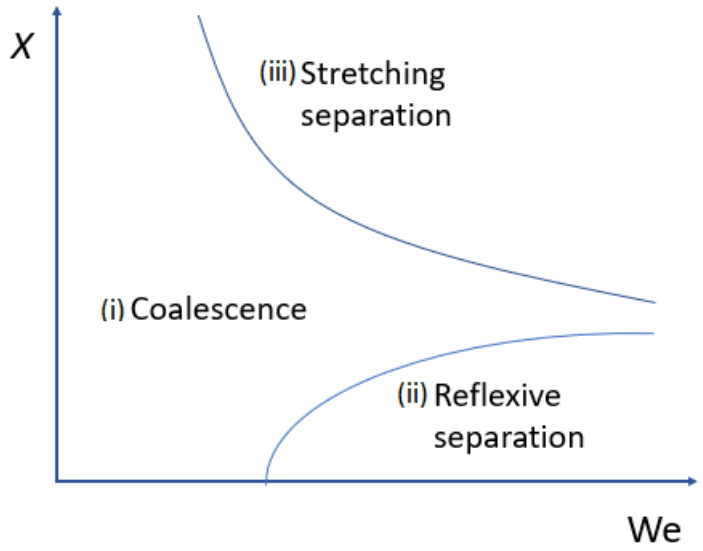


Figure 5.27: Collision regimes identified by Ashgriz & Poo [9] for binary water droplet collision of equal size.

run for a range of $Weber$ numbers and impact parameters X, each defined as

$$We = \frac{\rho U^2 D_s}{\sigma}, \tag{5.22}$$

$$X = \frac{2\mathcal{B}}{D_1 + D_2}, \tag{5.23}$$

where U is the impact velocity, D_s is the diameter of the smaller drop, $\mathcal{B}$ is the offset between the cell centres of the two droplets, and D_1 and D_2 are the diameters of each impacting droplet. The ratio between the two impacting droplets is defined as $\Delta = D_1/D_2$. Experiment configurations for this test can be found in detail in [9]. In this work, the following cases are tested;

Test	We	X	Δ
1	23	0.05	1
2	40	0.1	1
3	40	0.0	1
4	53	0.38	1
5	56	0.5	0.5

Table 5.6: Binary droplet collision parameters

The two colliding droplets are initialised just far enough from each other so as to not let them be in contact at $T = 0$. Each droplet is given a velocity such that their relative velocity is U. The initial velocity field is set to encompass the droplet plus a slight extension over the radius of the droplet to represent a fully developed air velocity around the droplets. The resolution shown for each test case denotes the lowest resolution with which CLSVOF-p can achieve results closest to the experiment.

Test 1: We=23, *X*=0.05

Binary droplet collision at this parameter (almost head-on collision) is indicated to be right at the boundary between coalescence and separation regimes by [9]. This test is a good indicator of the robustness of a scheme as less-accurate schemes may not be able to resolve the separation of the two droplets. At a relatively low We number, surface tension forces dominate so any shortcomings in the surface tension force estimation will be revealed in this test. CLSVOF-p and CLSVOF-a manage to capture the separation well using a resolution of only $d = 13$, which is 13 mesh cells across the diameter of one droplet.

Results in Fig. 5.29 show that CLSVOF-p produced the closest agreement to experimental results among the three schemes tested. The two hemispheres produced immediately after impact is captured best by CLSVOF-p. This result also seems the most

symmetrical of the three schemes which agrees with the experiment. A toroid is formed before the fluid mass reflexively separates away; this stage is again best captured by CLSVOF-p, with the interFoam result showing the least agreement. The final result of this collision is a complete reflexive separation with no satellite droplets formed. CLSVOF-p and CLSVOF-a both captured this end result, with CLSVOF-p having an advantage for the final shapes of the two droplets.

Fig. 5.28 show the schematic of the reflexive separation of two droplets which is reflected in the simulation results in Fig. 5.29.

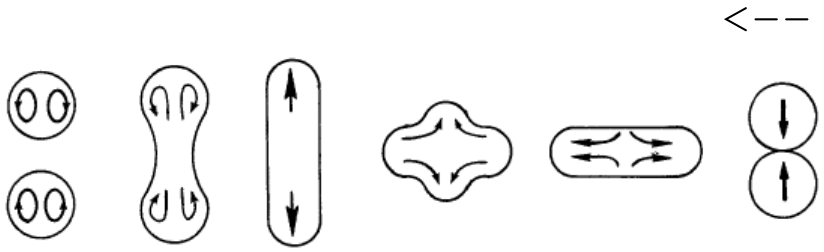


Figure 5.28: Schematic of reflexive separation for the collision of two equal-sized drops [9].

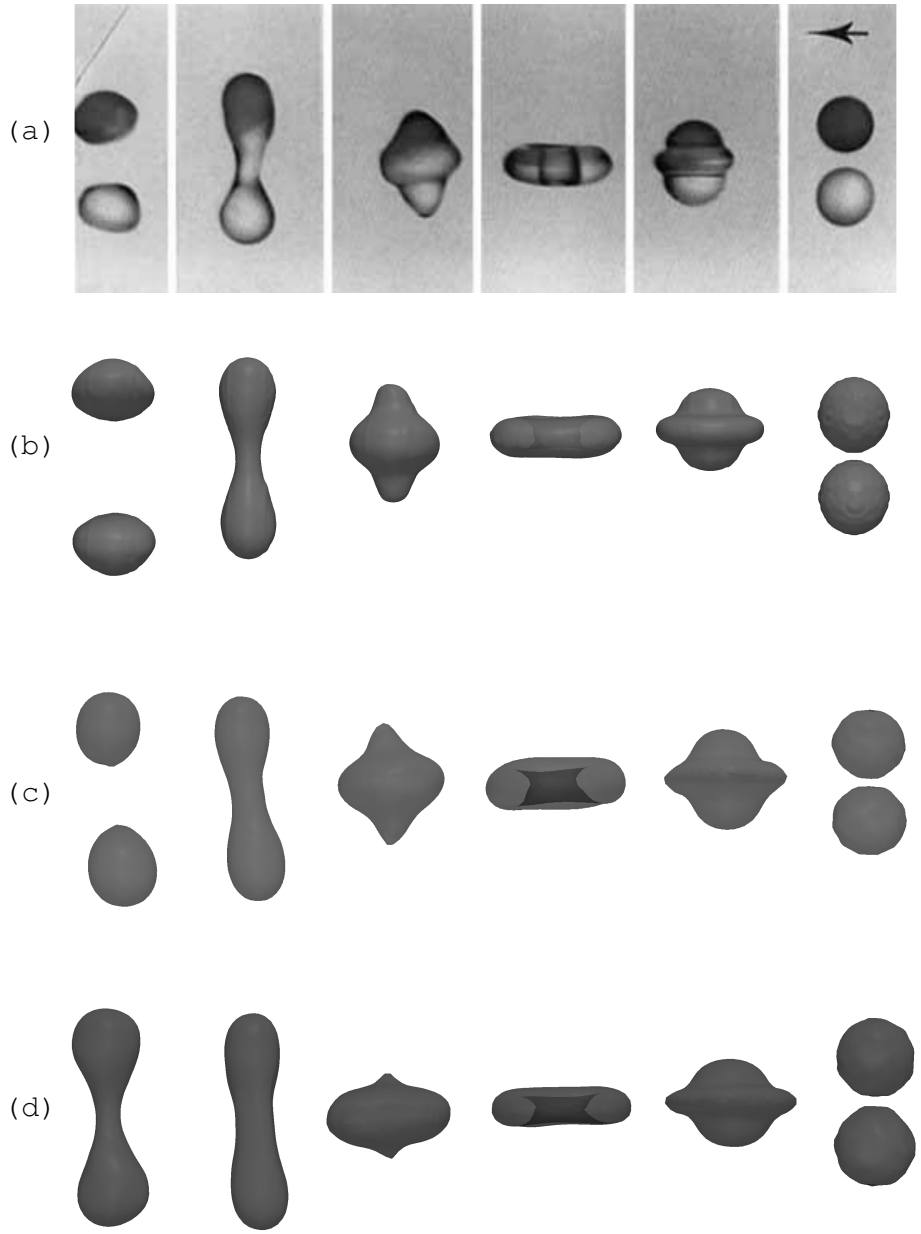


Figure 5.29: Test 1: Comparison between experimental result showing reflexive separation ((a) Fig.5 in Ashgriz and Poo 1990), numerical result using CLSVOF-p (b), CLSVOF-a (c), and using interFoam (d) at We=23 and x=0.05 using mesh $d = 13$. Note that the time evolution is from right to left.

Test 2: We=40, *X*=0.1

At a slightly higher We number and still a relatively low collision offset, this test is

expected to result in reflexive separation. Experiments by [9] show that a disc results after the collision, to form a four-lobed fluid mass as in Fig. 5.30. This mass is stretched as the two droplets try to separate, forming a liquid bridge between the two droplets. At this stage before the liquid bridge is formed, it can be seen that interFoam has produced some unphysical results, where a small oscillation appeared.

The liquid bridge is captured perfectly by CLSVOF-p, with CLSVOF-a trailing closely behind. interFoam has failed to approximate this liquid bridge. The final separation is also most accurately approximated by CLSVOF-p, with CLSVOF-a performing also reasonably well.

Test 3: We=40, *X*=0.0

The test parameters in this case is the same as in Test 2, but with no droplet offset i.e. this is a pure head-on collision. In this case it can be seen in Fig. 5.31 that CLSVOF-p has managed to simulate the end result of a reflexive separation with a resultant satellite droplet, most accurately compared to the experiment. The liquid bridge formed as the droplets separate is also captured by both CLSVOF-p and CLSVOF-a.

Test 4: We=53, *X*=0.38

This test parameter demonstrates a stretching separation that tends to occur at higher We numbers and higher offset X values. The end result of two separated droplets is captured by all three schemes, but only CLSVOF-p has managed to capture very accurately the stretched formation of the liquid mass before separation (Fig. 5.32).

Test 5: We=56, *X*=0.0

The binary collision between two unequal sized droplets is tested. A mesh of $D = 13$ across the diameter of the smaller droplet is used. In this case, CLSVOF-p and CLSVOF-a are shown to agree well with the experiment, with CLSVOF-p capturing the 'club' shape of the reflexive separation most accurately. Possibly due to issues in the calculation of curvature, interFoam managed to resolve the droplet separation,

but with the addition of a spurious satellite drop that resulted from the liquid column retracting improperly.

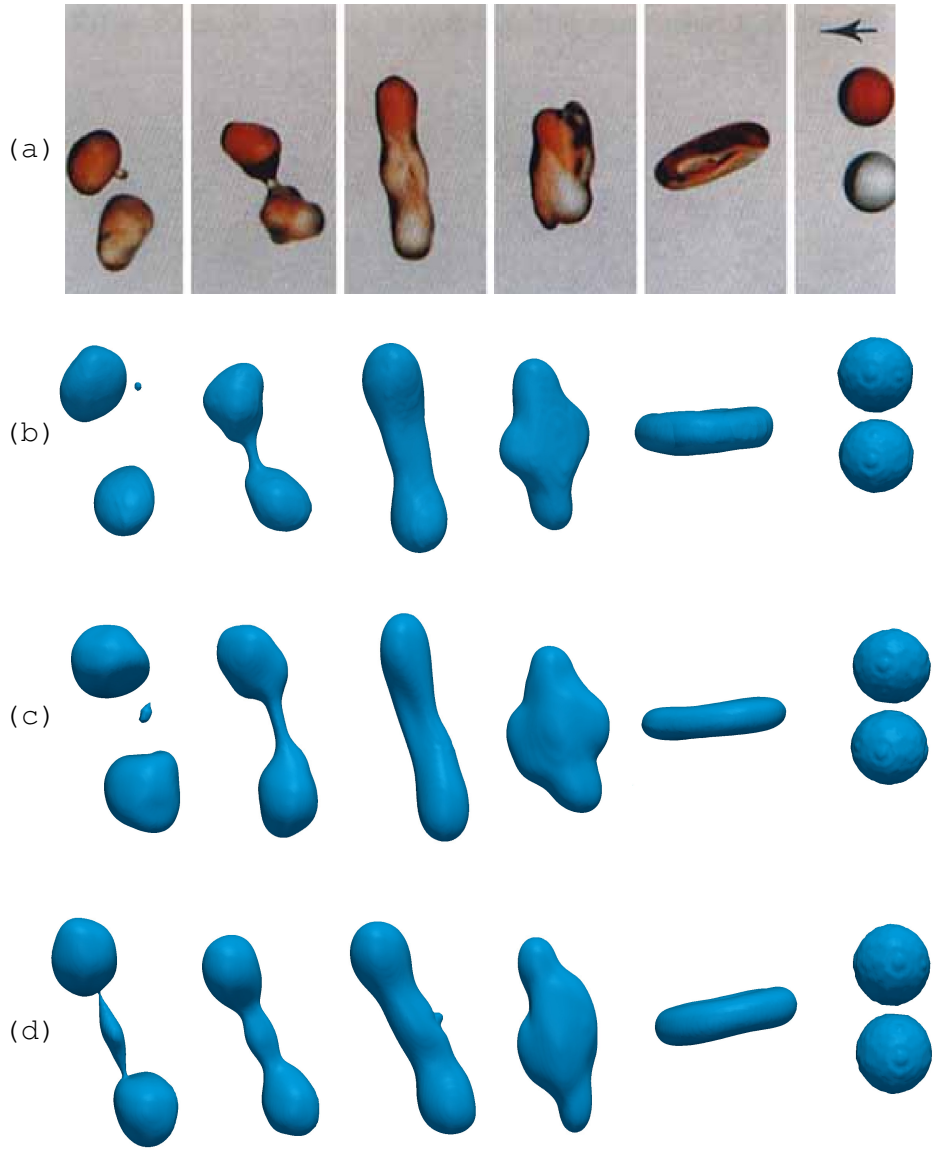


Figure 5.30: Test 2: Comparison between experimental result showing reflexive separation ((a) Fig.10 in Ashgriz and Poo 1990), numerical result using CLSVOF-p (b), CLSVOF-a (c), and using interFoam (d) at We=40 and x=0.1 using mesh $d = 24$. Note that the time evolution is from right to left.

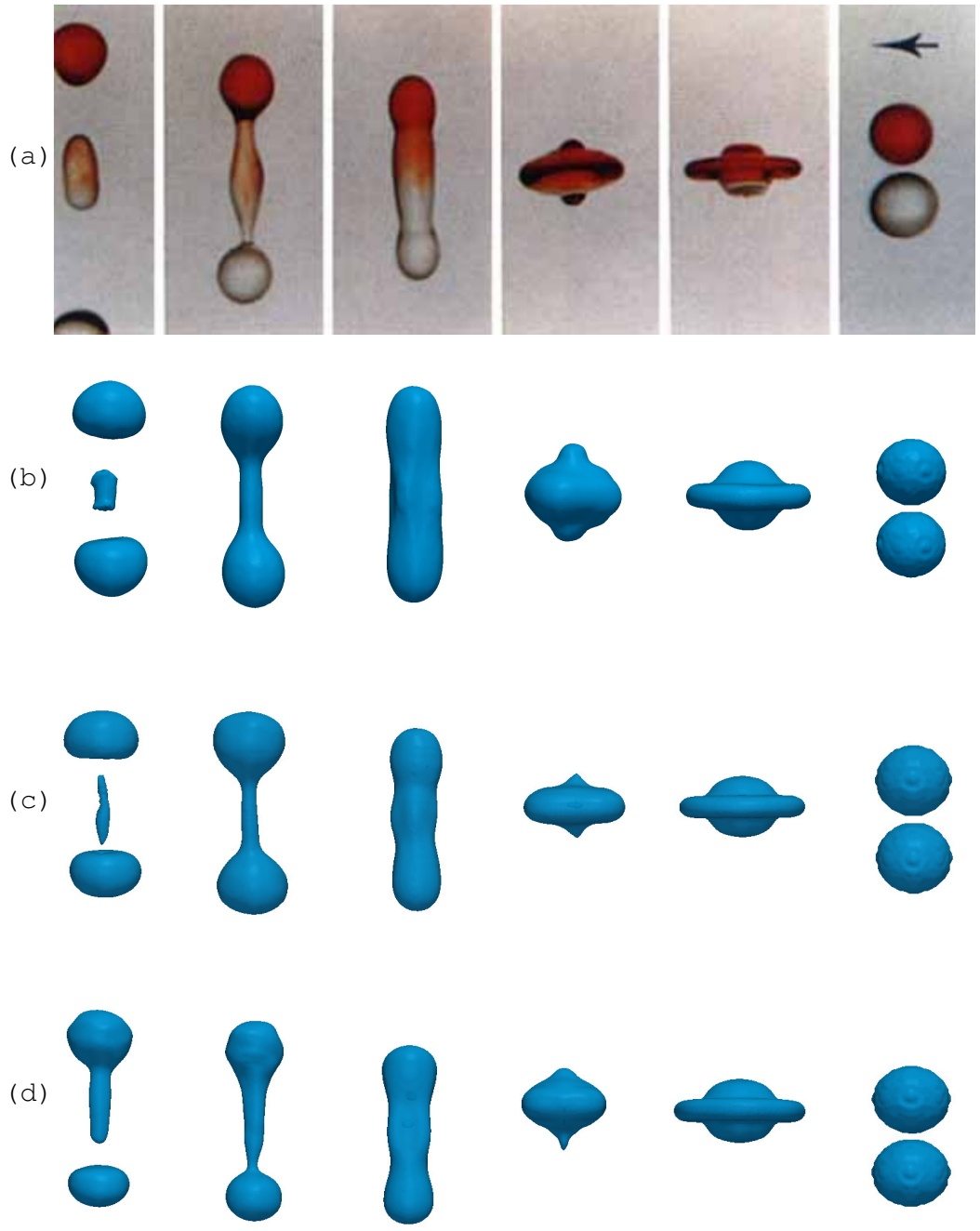


Figure 5.31: Test 3: Comparison between experimental result showing reflexive separation ((a) Fig.6 in Ashgriz and Poo 1990), numerical result using CLSVOF-p (b), CLSVOF-a (c), and using interFoam (d) at We=40 and x=0.0 using mesh $d = 24$. Note that the time evolution is from right to left.

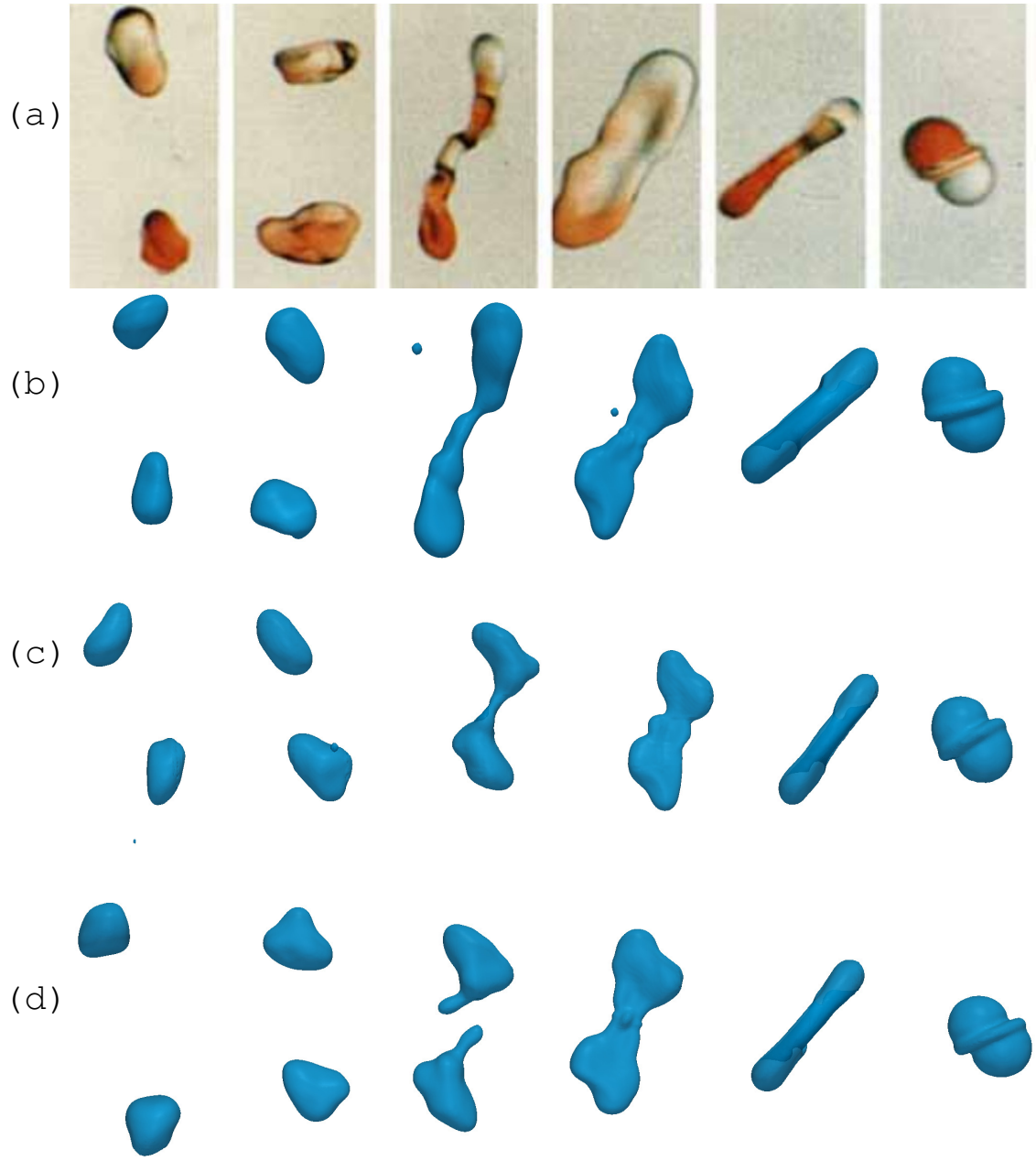


Figure 5.32: Test 4: Comparison between (a) experimental result showing stretching separation (Fig.12 in Ashgriz and Poo 1990), numerical result using (b) CLSVOF-p, (c) CLSVOF-a, and (d) using interFoam (bottom) at We=53 and x=0.38 using mesh d=24h. Note that the time evolution is from right to left.

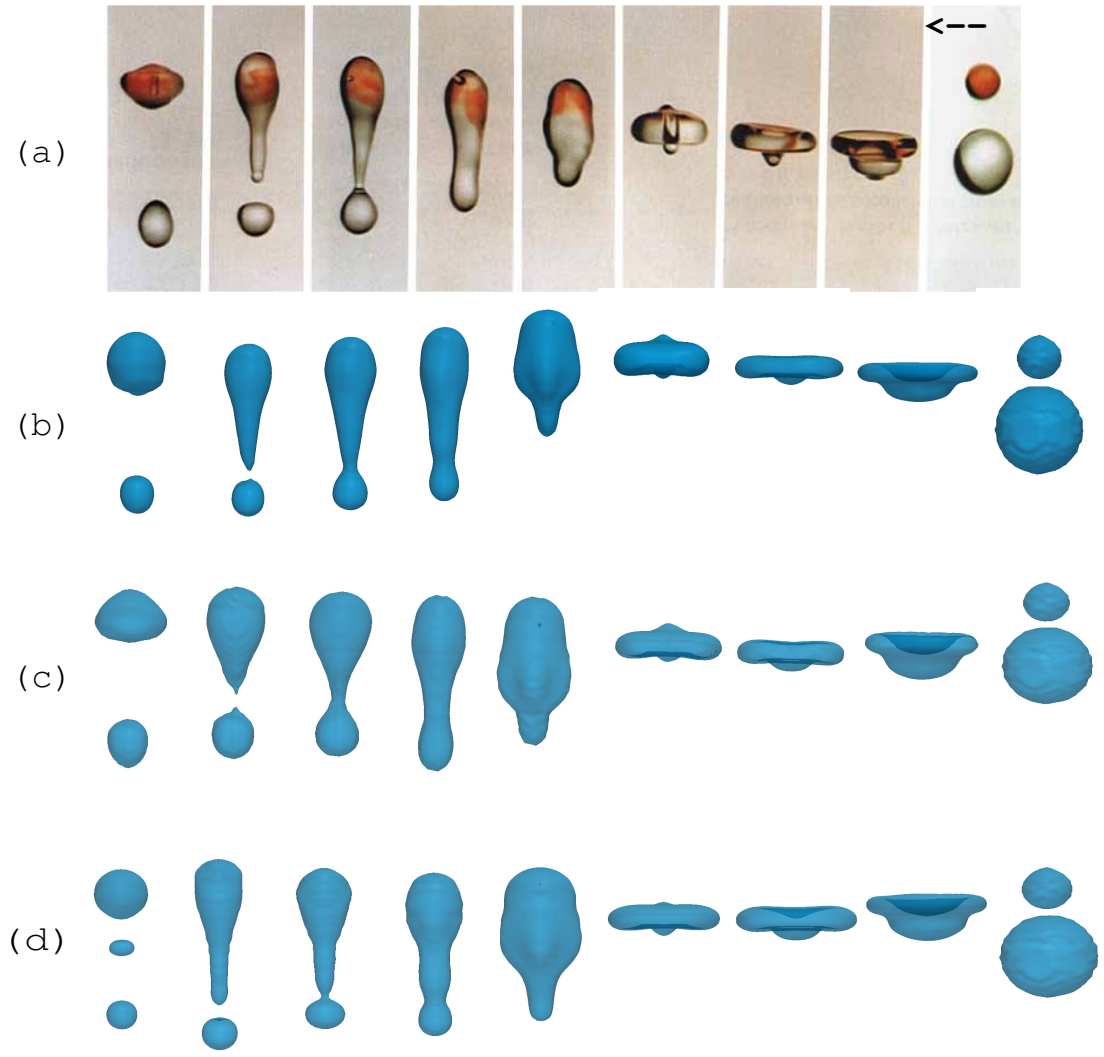


Figure 5.33: Test 5: Comparison between (a) experimental result showing reflexive separation (Fig.20 in Ashgriz and Poo 1990) of unequal sized drops at $\Delta = 0.5$, numerical result using (b) CLSVOF-p, (c) CLSVOF-a, and (d) interFoam at We=56 and x=0 using mesh $d_1 = 13$. Note that the time evolution is from right to left.

5.3.9 Liquid jet in gaseous crossflow

The study of liquid jets in a cross-flow or transverse jets is one of great interest as injecting liquids into a fast-moving gaseous cross-flow is a common method to atomise the liquid. This has many applications within industry wherever a liquid surface-area to volume ratio is desirable; in fuel sprays, this will ensure optimal mixing with air to ensure an even combustion [132].

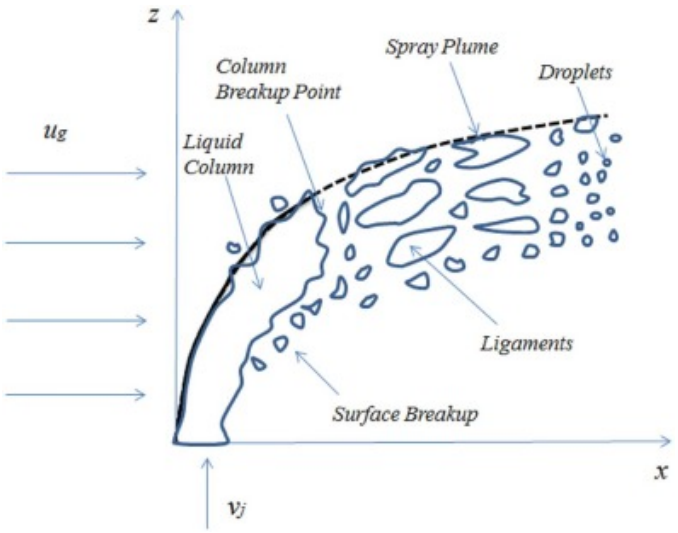


Figure 5.34: Schematic of a jet penetrating into a crossflow displaying the structures in a jet breakup. Reprinted from Wang et al. [10] .

Fig. 5.34 shows the liquid jet exiting the nozzle as a column, which starts to 'ruffle' from axial instability, and then breaks into ligaments and droplets [133]. As liquid is stripped away from the surface of the jet and surface breakup gains dominance, the penetration height of the liquid decreases [15]. As the fragments of the liquid jet undergo secondary breakup which produces smaller droplets, a spray is formed.

There are various characteristics used to describe the behaviour of a liquid jet in cross-flow. Among these are the breakup regimes of the jet, the trajectory and penetration of the liquid, the formation of droplets, the minimum size of droplets, and breakup length. The parameters used to control the study of jet in cross-flow are often dimensionless.

The most commonly used ones are the cross-flow $Weber$ number We_G, the jet $Weber$ number We_L, the velocity of the jet u_L, jet diameter d_{jet} and the momentum flux ratio q;

$$We_G = \frac{\rho_G u_L^2 d_{jet}}{\sigma}, \tag{5.24}$$

$$We_L = \frac{\rho_L u_L^2 d_{jet}}{\sigma}, \tag{5.25}$$

$$q = \frac{\rho_L u_L^2}{\rho_G u_G^2}. \tag{5.26}$$

However, numerous other parameters have also been used to investigate this phenomenon, such as ambient pressure, the jet/cross-flow viscosity ration, and nozzle geometry. This wide array of variables render the phenomenon extremely complex and time-consuming to study.

The simulation of liquid jets in cross-flow is also numerically challenging. The complex interfacial structures that is expected during liquid breakup require a robust two-phase solver that is able to retain the fine structures that occur during breakup [134]. Access to high-performance computing facilities is also fundamental to produce meaningful simulations that are able to capture the finer structures of the breakup.

In this work, a liquid jet in a gaseous cross-flow of $We_G = 40$ is simulated to show the viability of the proposed CLSVOF-p for highly complex simulations. Following the success of the CLSVOF-p method in simulating the phenomena of binary droplet collision in Section (5.3.8), which is a phenomenon fundamental for the study of jet atomisation, the proposed method can now be tested on a larger scale.

The parameters used in this simulation are taken from [135]. The liquid jet in injected into a fully-developed cross-flow of pressure $100,000\ Pa$ in the domain illustrated in Fig. 5.35(a), where the jet inlet is located 0.002 m downstream from the cross-flow

inlet boundary at $x = 0$. A no-slip boundary condition is imposed on the $y = 0$ plane. Outflow boundary conditions are imposed on the rest of the boundary planes.

The simulation time-step criterion is a variant of that presented in [135], based on the CFL criterion;

$$\Delta t < \frac{1}{2}\frac{h}{|u_G|}, \tag{5.27}$$

where $h = min(\Delta x, \Delta y, \Delta z)$ of the smallest mesh in the domain. Fig. 5.35(b) shows the mesh refinement region to ensure the only the region where the jet is expected to penetrate into is refined. The refinement is performed using five levels of refinement from the largest unrefined cell, with three specific refinement regions $M1 = 5 \times 10^{-5}$ m , $M2 = 10 \times 10^{-5}$ m, and $M3 = 20 \times 10^{-5}$ m. The mesh (Fig. 5.9) is composed of 99% hexahedra and 1% polyhedra of various types, generated using the OpenFOAM utility $snappyHexMesh$. The jet reaches full penetration at $t = 0.02s$. The simulation is carried out on the High Performance Computing Wales system on 2 x Intel® Xeon® Gold 6148 CPU (2.40 GHz) on a total of 40 cores, taking 15 days to complete.

Table 5.7: Material properties for the liquid jet in cross-flow simulation

ρ_L (kg/m^3)	ρ_G (kg/m^3)	μ_L (kg/ms)	μ_G (kg/ms)	g (m/s^2)	σ (kg/s^2)
997	1.18	8.94×10^{-4}	1.86×10^{-5}	-9.81	0.0708

Table 5.8: Parameters of the liquid jet in cross-flow simulation

We_L	We_G	$U_L(ms^{-1})$	$U_G(ms^{-1})$	q	Re_G
3471	40	17.7	54.8	88.2	2781

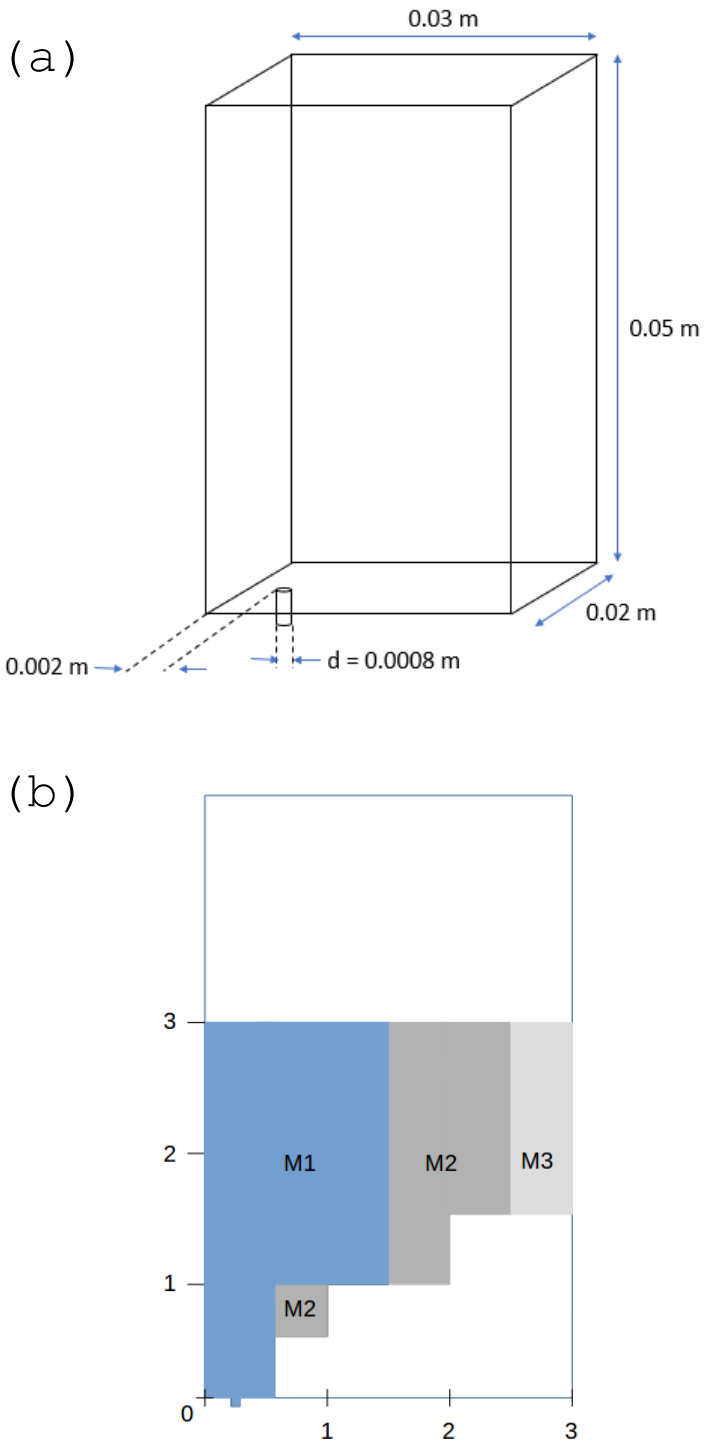


Figure 5.35: Computational domain of jet in cross-flow test showing (a) the domain setup and (b) the staggered mesh refinement regions $M1$, $M2$, and $M3$.

Table 5.10: We_G breakup transitions of round liquid jets in crossflows obtained by Mazallon [16] and Sallam [17].

Source	Mazallon et al.	Sallam et al.
Column/bag breakup transition	We_G =5	We_G =4
Bag/multimode breakup transition	We_G =60	We_G =30
Multimode/shear breakup transition	We_G =110	We_G =110

Table 5.9: Mesh details of the jet in cross-flow simulation showing the number of cells of each type.

Hexahedra	Polyhedra					Minimum cell volume	Maximum cell volume
	6 faces	9 faces	12 faces	15 faces	18 faces		
19,502,725	7538	166,783	3446	942	65	1.76×10^{-13}	7.23×10^{-10}

Mazallon et al. [16] and Sallam et al. [17] suggested that for round non-turbulent liquid jets in a cross-flow, three breakup regimes can be classified according to the crossflow Weber number We_G (Table 5.10). A liquid jet in a cross-flow with the parameters in this work at $We_G = 40$ is expected to undergo multimode breakup, which is a complex mixture of bag and shear breakup regimes. The results of the simulation at full penetration are shown in Figs. (5.36 - 5.39). Fig. 5.36 shows the profile view of the liquid jet with secondary droplets visible up to the point where the mesh is refined.

Fig.(5.37) shows ligaments forming off the edge of the jet column as a result of shearing. The ligaments then fragment into droplets. These droplets then continue on downstream to interact with the cross-flow, mixing with the gaseous phase. A considerable amount of detail is captured as can be seen in Fig. 5.38 considering the size of the mesh. Ligaments or fluid parcels can be seen to form in the early stages of the breakup; as

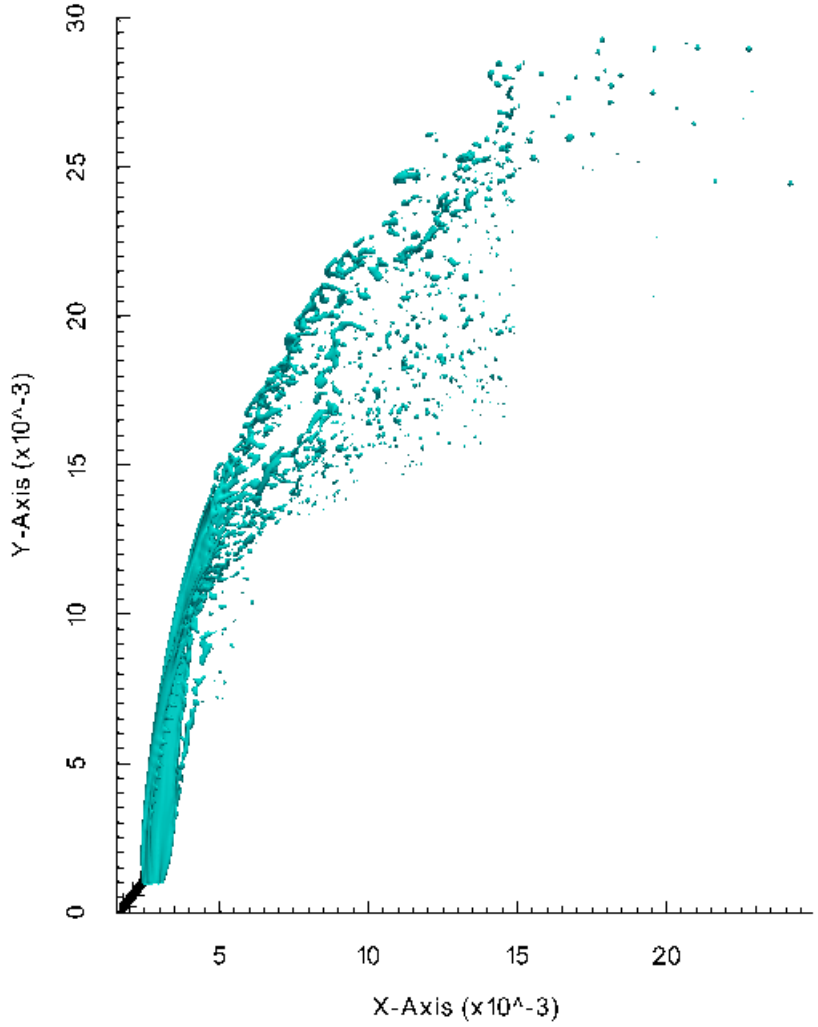


Figure 5.36: Jet in gaseous cross-flow at full penetration, profile view

they move further downstream in the crossflow direction, the ligaments then undergo secondary breakup into droplets. Liquid jets in a crossflow using the parameters in this work is in the multimode breakup regime where 'bags' (liquid membranes of the concave structures immediately before breakup) are expected to form. However, this structure is not captured in this simulation due to the mesh resolution used. The same issue is also faced by [135] where a mesh size of $h = 5.68 \times 10^{-7}$ m is used for the liquid region.

The velocity magnitude profile is shown Fig. 5.40 demonstrates the vortices that appear

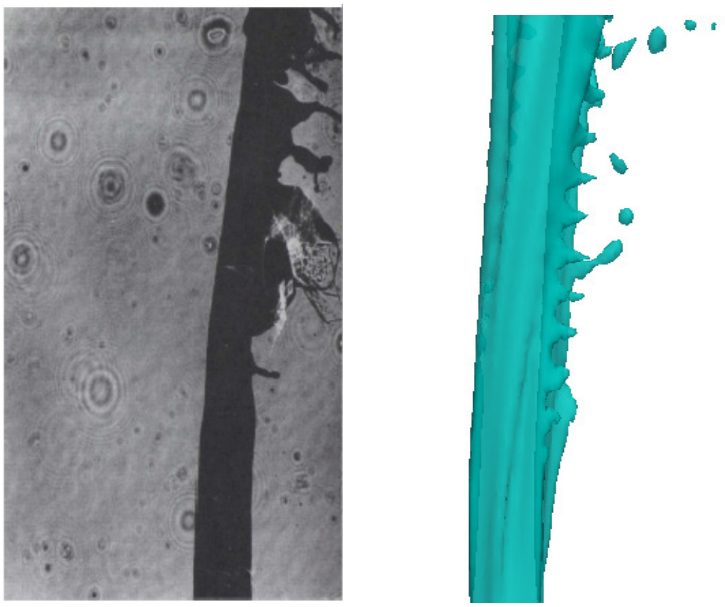


Figure 5.37: Multimode breakup regime from experiment by Ashgriz [11] compared with the simulation in this work.

downstream of the crossflow which is important to ensure optimal mixing of the fluid and gaseous phases. The wake region is shown to go upwards with the motion of the jet. Higher velocities are observed in the lower region behind the wake due to the acceleration of the flow.

Cross-sections along the $y-axis$ are taken to demonstrate instantaneous velocity magnitude and pressure in Figs. (5.41 - 5.43). The crossections showing the pressure at various heights along the $y-axis$ in Figs. (5.41(b), 5.42(b), 5.43(b)) demonstrate a high-pressure zone forming on the upwind side of the jet (with respect to the crossflow) and a low pressure wake region on the downwind side due to the aerodynamic blocking of the jet column.

The crossections showing velocity magnitude in Figs. 5.41(a), 5.42(a), 5.43(a) show that the liquid jet column causes a shear layer to grow from it. As it moves downstream of the jet, the shear layer grows in size as the progression shown from the jet inlet at Fig. 5.41(a) to the downstream section of the jet at Fig. 5.43(a). The beginning of a

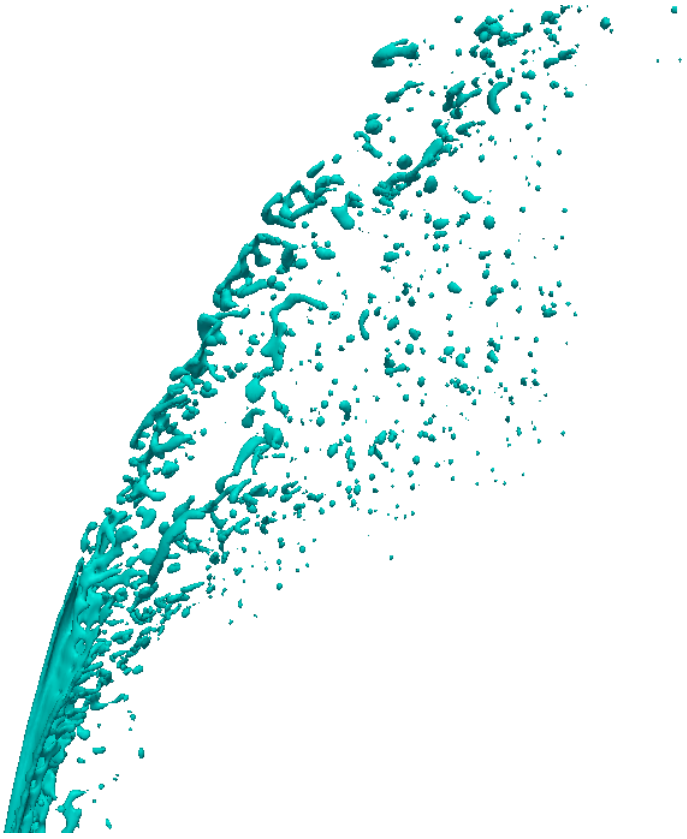


Figure 5.38: Close-up view of the breakup region of the liquid jet

Karman vortex street can also be detected originating from the jet which acts as a blunt body, but due to the low mesh refinement level downstream of the crossflow, this is not fully resolved.

The trajectory and penetration of the liquid jet are important parameters in a typical liquid jet in a crossflow study [136] [137]. The flowfield is usually divided into two regions; the region of the jet with little to no deflection of the column, and the wake region where the jet is almost parallel to the crossflow [133]. The trajectory of the jet determines how quickly the jet deflects and reaches an almost parallel flow with the crossflow. Many power law correlations have been produced for the evolution of the jet trajectory, commonly in the form of

$$\frac{y}{d_{jet}} = c_1 q^{p1} \left(\frac{x}{d_{jet}} \right)^{p2} \tag{5.28}$$

where c_1 is a constant, $p1$ and $p2$ are power exponents, and q is the momentum flux

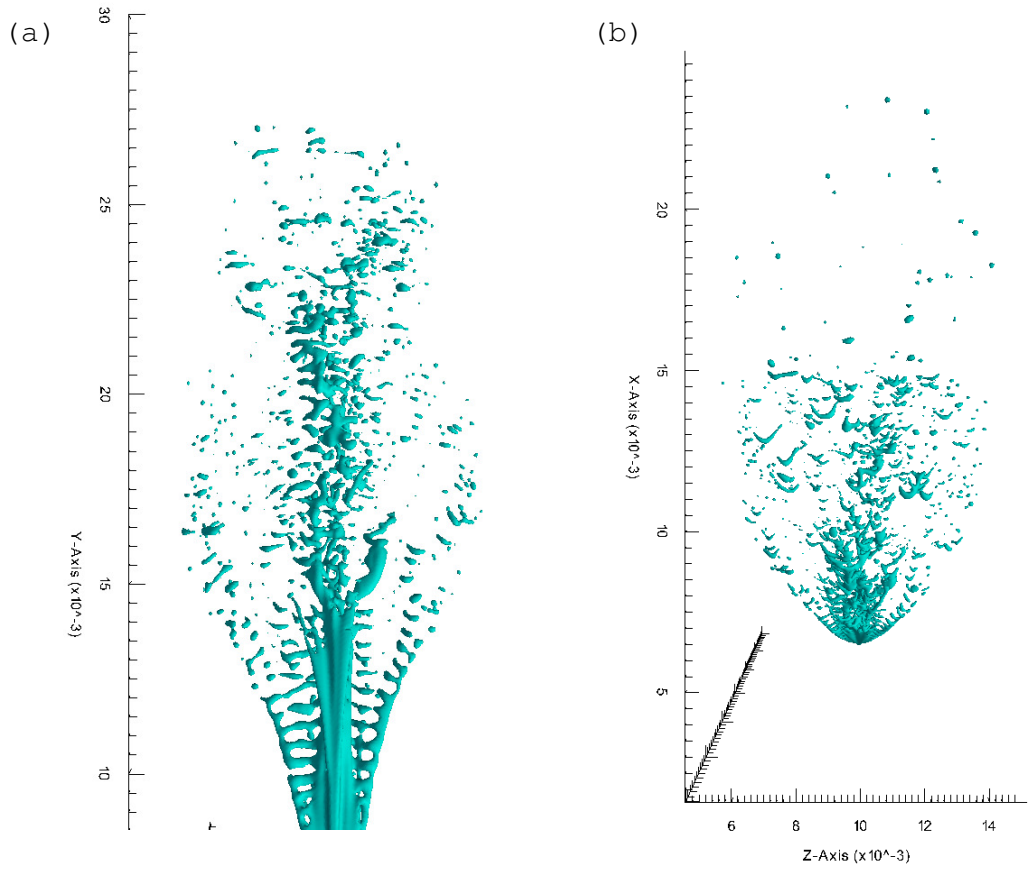


Figure 5.39: Jet in gaseous cross-flow at full penetration from (a) front view and (b) top view.

ratio. In all correlations the height of the jet and the length of the jet are normalised by the jet diameter as y/d_{jet} and x/d_{jet} respectively. The momentum flux ratio is considered a main influencer of the trajectory as it is considered in all of these correlations obtained for various liquids under various temperatures and pressures; [138], [15], [139], [140], [141]. [142], [143], [12], [14], [144], [145], [146], [147], [13]. A comprehensive summary of the different correlations and the range of test conditions under which they were drawn from can be found in [133].

The trajectory of the liquid jet obtained in this work is compared with the correlations produced under similar conditions. Where a liquid other than water is used, the viscosity ratio is accounted for to ensure a comprehensive correlation, except in the case of

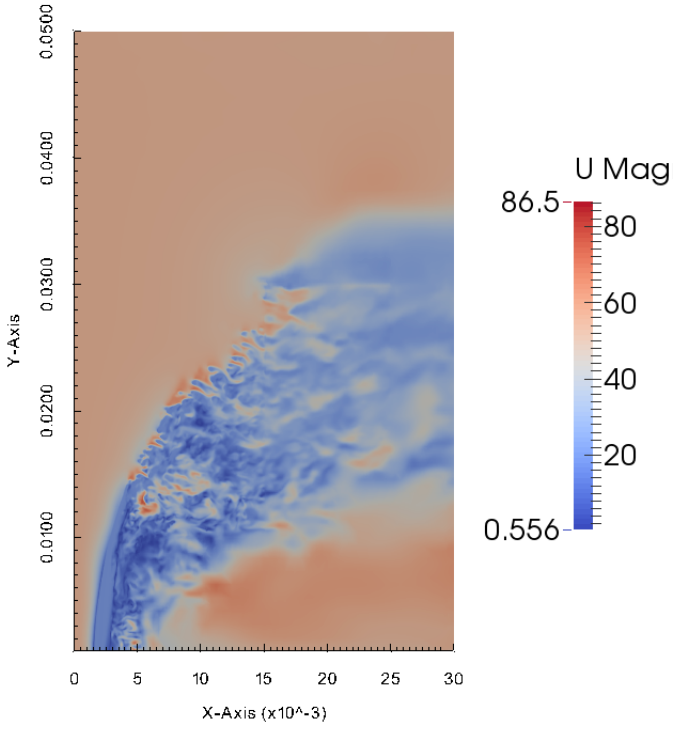


Figure 5.40: Velocity magnitude shown in profile view

Bolszo [15].

Table 5.11: Liquid jet trajectory references for a subsonic gaseous crossflow at standard temperatures and pressures test conditions.

Reference	q	We_G	x/d_{jet}
Iyogun [12]	8-726	9-159	0-64
Farvardin [13]	10-135	28-82	0-50
Birouk [14]	8-726	9-159	0-40
Bolszo [15]	38,136	17,145	0-70

Iyogun:

$$\frac{y}{d_{jet}} = 1.997q^{0.444}\left(\frac{x}{d_{jet}}\right)^{0.444} \tag{5.29}$$

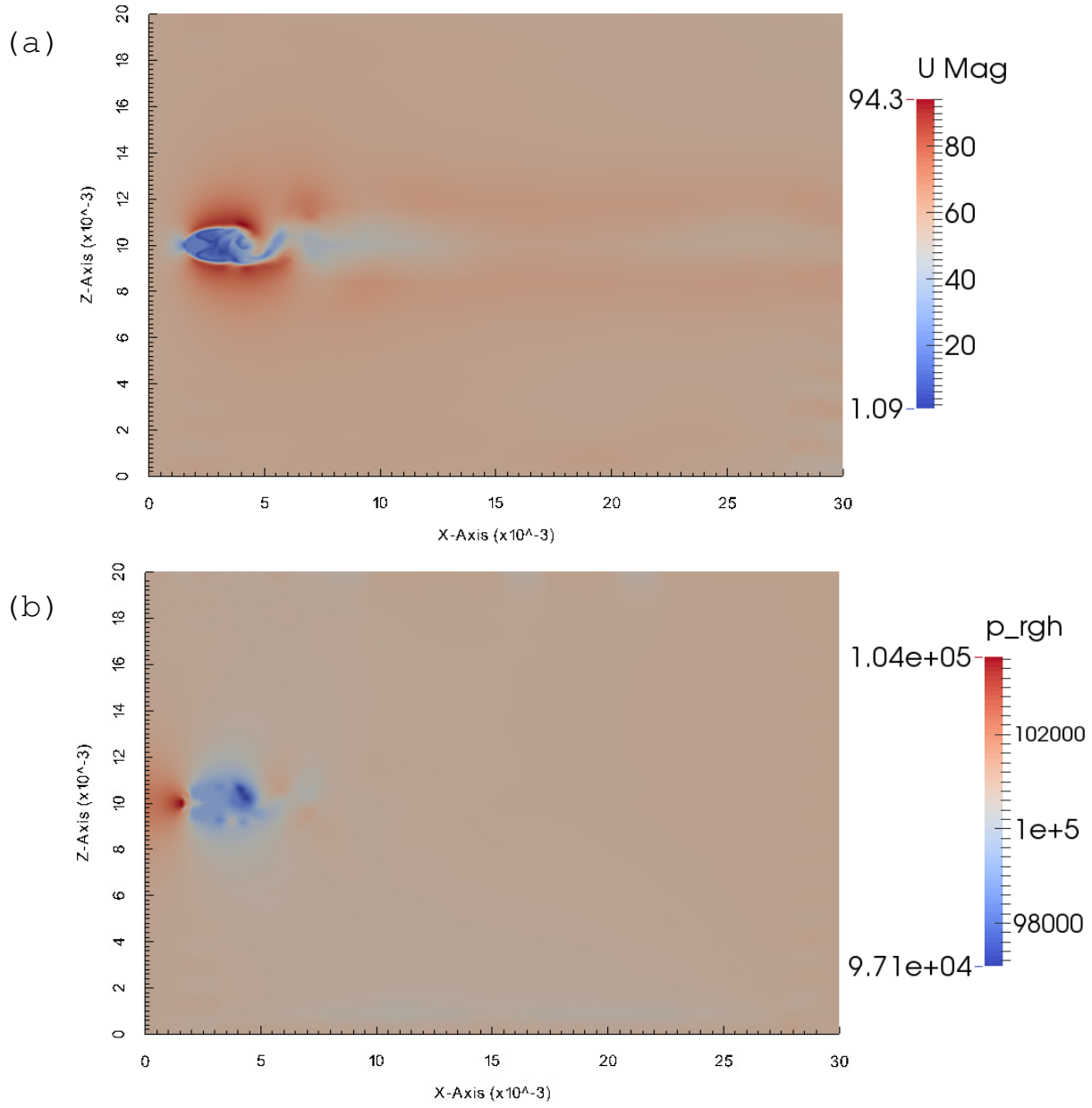


Figure 5.41: (a) Velocity magnitude and (b) pressure, both at $t = 0.02$ s and plane $y = 0.0017$.

Farvardin:

$$\frac{y}{d_{jet}} = 3.688q^{0.43}\left(\frac{x}{d_{jet}}\right)^{0.384} We_G^{-0.085}\left(\frac{\mu_{jet}}{\mu_W}\right)^{-0.222} \tag{5.30}$$

where μ_W is the viscosity of water taken to be $8.943 \times 10^{-4}\ kg\ m^{-1}s^{-1}$.

Birouk:

$$\frac{y}{d_{jet}} = 1.627q^{0.47}\Big(\frac{x}{d_{jet}}\Big)^{0.46}\Big(\frac{\mu_{jet}}{\mu_W}\Big)^{-0.0.079} \tag{5.31}$$

Bolszo:

$$\frac{y}{d_{jet}} = 1.48q^{0.43}\Big(\frac{x}{d_{jet}}\Big)^{0.43} \tag{5.32}$$

Fig. 5.44 shows a similar trend between the profile obtained by the simulation at full penetration with Farvardin's empirical model at the beginning of the deflection. However as the flow develops, the simulation moves closer to the models given by Iyogun and Birouk. The correlation by Bolszo underpredicts the trend for all segments of the flow; while the momentum flux ratio is indeed accounted for as well as the range of valid We_G values, the model is built on water-in-oil emulsion jets. Since the model by Bolszo did not include liquid viscosity in their expression, this accounts for the discrepancy and confirms that the trajectory of a liquid jet in a crossflow is dependent on more than just the two most-often used parameters We_G and q.

5.4 Conclusion

The proposed solver is verified against fluid flows with surface tension. Two methods of calculating mean curvature is implemented for CLSVOF, where CLSVOF-p uses the LS field to calculate curvature, and CLSVOF-a uses the VOF field. The performance of these two implementations are compared against that of interFoam and where applicable, interFlow. Results indicate that CLSVOF-p generates the least amount of spurious currents. In tests with reference solutions (2D rising bubble, Rayleigh-Taylor instability, droplet impact with dynamic contact angle), CLSVOF-p appears to

be quantitatively the most accurate out of all the compared schemes. In a test where surface tension does not dominate (2D dam break), all tested schemes perform similarly with CLSVOF-p, CLSVOF-a, and interFlow showing sharper interfaces than interFoam.

In more complicated tests such as the droplet splashing on a dry surface and on fluid, CLSVOF-p shows the greatest qualitative agreement to experiment images, where structures such as finger-like projections are captured successfully. Finally, the phenomena of a liquid jet in a cross-flow and the binary collision of droplets under standard temperatures and pressures are simulated. The results obtained using CLSVOF-p for the binary collision investigation agree well with experimental findings even with relatively low mesh resolution and are better compared to interFoam. In the simulation of a liquid jet in a gaseous crossflow, the result obtained by CLSVOF-p demonstrated a fair agreement for the evolution of the jet column to several empirical models. Jet breakup structures such as surface breakup, liquid ligaments, and droplets are captured by the simulation.

The findings from this chapter successfully demonstrated the capability of the proposed CLSVOF solver in solving two-phase fluid flow cases with complicated flow structures and surface tension effects.

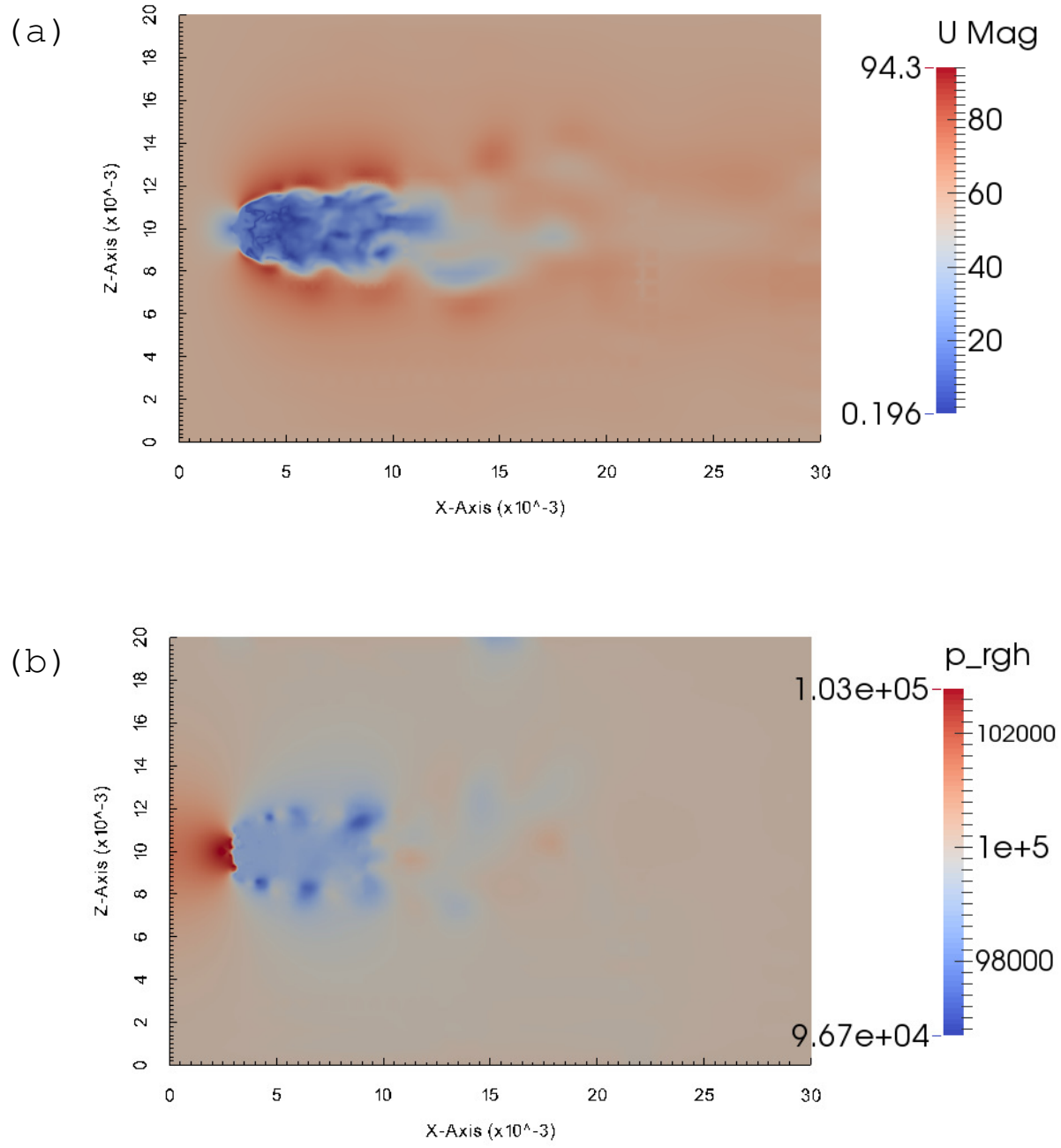


Figure 5.42: (a) Velocity magnitude and (b) pressure, both at $t = 0.02$ s and plane $y = 0.0086$.

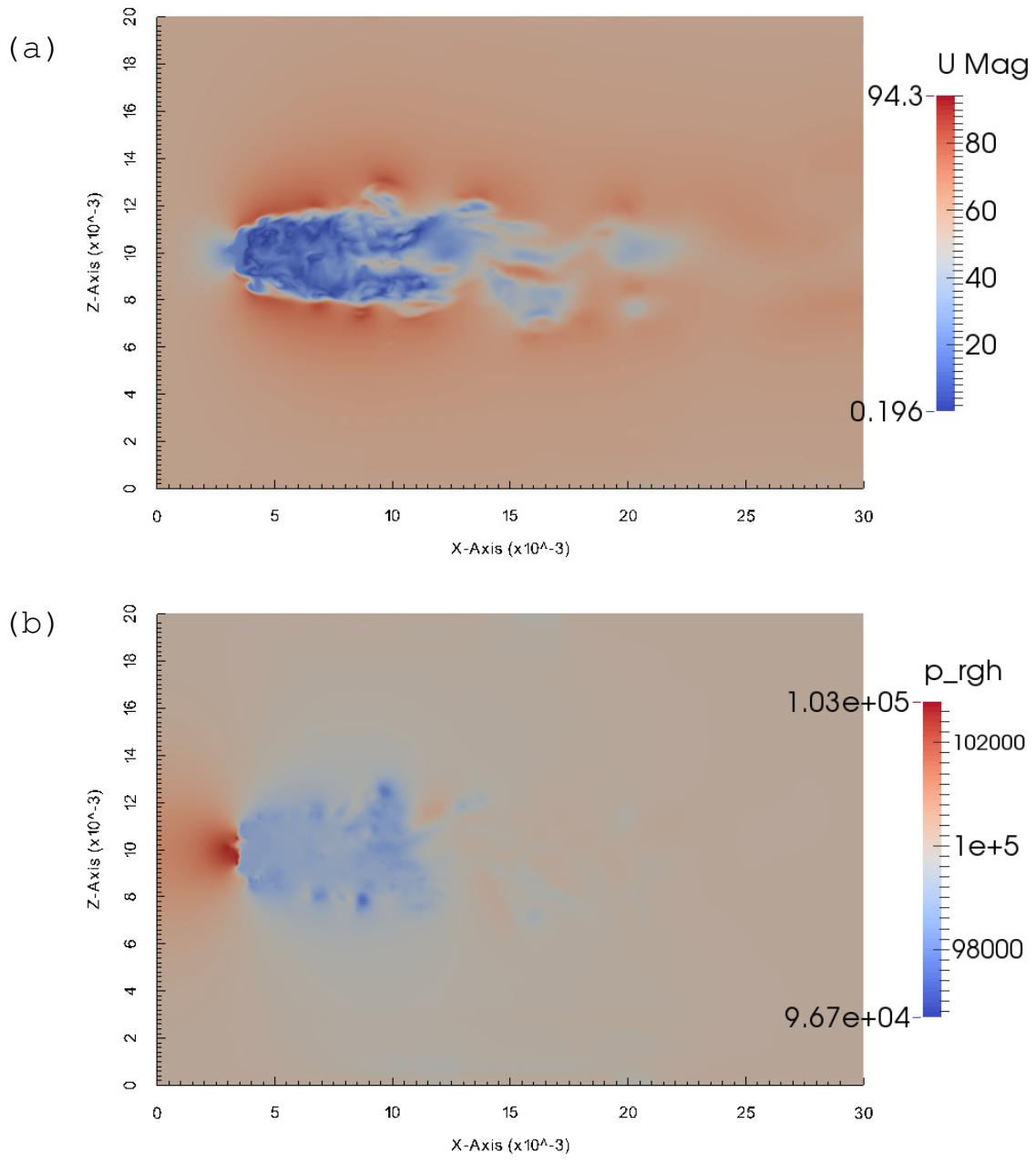


Figure 5.43: (a) Velocity magnitude and (b) pressure, both at $t = 0.02$ s and plane $y = 0.01$.

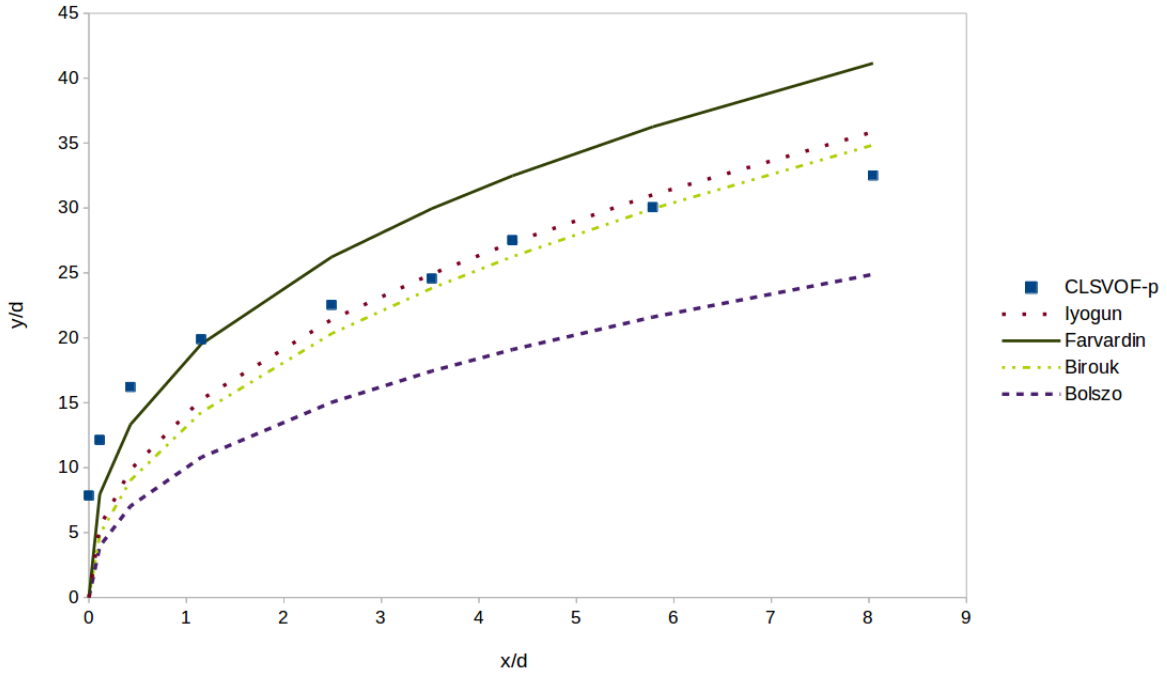


Figure 5.44: Jet profile at full penetration compared against empirical correlations by [12], [13], [14], [15].

Chapter 6

Conclusion and Outlook

The work accomplished in this thesis was divided into two parts; the development and implementation of a hyperbolic equation solver to handle discontinuous solutions, and the development and implementation of the CLSVOF scheme in OpenFOAM for two-phase incompressible flows followed by validation and numerical simulation of complex fluid flows phenomena including phenomena related to jet atomisation.

6.1 Conclusion

The first part of this work proposed a new hyperbolic partial differential equation solver based on multi-moment methods, of the CIP-CSL family. This addressed the first objective as outlined in Chapter 1, which is to develop a hyperbolic partial differential equation solver that is able to capture discontinuities and smooth solutions simultaneously well with minimal numerical oscillation and diffusion.

A scheme called CIP-CSL3U was proposed to be used together with an existing scheme, CIP-CSL3D. Using an ENO-like formulation, CIP-CSL3U and CIP-CSL3D are combined into CSL3DU and CSL3ENO; while they both use the same constituent stencils, the way the stencils are selected during runtime differs. Two smoothness indicator for the stencils were proposed, one each for CLS3DU and CSL3ENO. Both of the proposed schemes work well on benchmark tests, outperforming CIP-CSL3CW, CIP-CSL2, and CIP-CSL3D and CIP-CSL3U individually. It was found that CSL3DU and

CSL3ENO are both high-order accurate (4th order), with most numerical oscillation being successfully suppressed without giving way to numerical diffusion.

The second part of this work focused on the improvement of the two-phase incompressible flow solver in OpenFOAM, addressing the second, third, and fourth objectives outlined in Chapter 1.

The contributions from this part of the work are the major modifications and extensions upon existing open source code in order to make improvements to the interface reconstruction implementation. This was done by implementing a fully 3D geometric level set (LS) method to couple with the most recent volume of fluid (VOF) addition to OpenFOAM, interFlow. This is a novel combination where an explicit geometric interface reconstruction using the Level Set method is coupled with an advection scheme that tracks the interface *within* a single time-step.

An interface reconstruction algorithm based on cell tetrahedralisation was implemented to calculate the exact volume of fluid residing under a plane in a general convex polyhedral cell. This procedure was then described in great detail to improve the accessibility of this work.

The proposed scheme was then validated using structured and unstructured meshes in 2D and 3D. For scalar transport problems, CLSVOF was shown to outperform interFoam in all tests and interFlow in most tests. Where the effects of surface tension are negligible such as in the dam-break test (Section 5.3.2), all schemes produced visually similar results. However, for other fluid flow simulations, CLSVOF consistently outperformed interFoam and interFlow, highlighting the importance of a good estimation of the surface tension force to ensure the robustness of a two-phase incompressible flow solver.

Fluid structure detail such as crown formation in droplet splashing that was not captured by interFoam and interFlow was captured by CLSVOF, as seen in Section 5.3.7. This is due to the improved curvature calculation in CLSVOF leading to a better es-

timation of the surface tension force. Quantitatively, this improvement in curvature calculation was captured in the static droplet at equilibrium test in Section 5.3.1, where it is shown that results obtained using CLSVOF-p (where the curvature is calculated using the Level-Set field) yielded the least amount of spurious currents at the interface. The implementation of the Yokoi Dynamic Contact Angle model enabled the simulations carried out in Section 5.3.5 and Section 5.3.6. The results obtained from these simulations show that qualitatively, CLSVOF-p showed the greatest agreement with experimental findings compared to CLSVOF-a and interFoam and also showed greater detail in the form of finger-like projections around the spreading edge of the droplet. It is also worthwhile to note that in many of the surface tension dominant flows simulated in this work, the interFlow solver did not manage to produce a physical solution. This is due to the fact that the interFlow solver had been conceived for use in marine science and engineering, where the effects of surface tension can usually be considered irrelevant.

CLSVOF was also shown to predict the outcomes of various binary droplet collisions accurately, as seen in Section 5.3.8. CLSVOF-p was shown to generally produce the best agreement when compared to images obtained experimentally. Finally, the simulation of a liquid jet in a gaseous crossflow using CLSVOF-p in Section 5.3.9 showed some surface breakup detail and the jet trajectory displayed reasonable agreement with empirical models.

It is therefore concluded that the proposed implementation of combining an explicit interface capturing method on the VOF solvers in OpenFOAM with the Level Set method yielded a marked improvement over the 'stock' two-phase incompressible flow solver available in OpenFOAM. Its success in various benchmark tests and in some complex fluid flow applications indicate that the proposed method has potential to work well in various other incompressible two-phase flow applications.

6.2 Outlook

The proposed CIP-CSL3DU and CIP-CSL3ENO methods have been shown to suppress oscillation whilst maintaining profile sharpness at discontinuities. However, the numerical oscillations have not been completely eliminated. A possible improvement might be achieved by including another stencil in the selection, one that suppresses oscillations completely. A different smoothness indicator might be implemented, along with a different selection mechanism.

For the CLSVOF implementation, some of the possible room for improvement may include a more efficient polyhedra tetrahedralisation process that require fewer steps to achieve the required level of decomposition. This may help reduce the calculation costs. To increase the possible applications of this scheme, a heat transfer model may also be implemented, as well as a provision for the handling of dynamic mesh refinement. This may be particularly useful for cases such as liquid jet atomisation where computational costs are high. Moreover, the current implementation of the dynamic contact angle model requires specific parameters for specific cases; a more generalised contact angle model may be implemented to work on a wider range of cases without needing experimental data. Furthermore, the current improvement was implemented in OpenFOAM 2.1.x. It may be of interest to implement it on newer versions to see if any further improvements can be made.

Appendix

Pressure-Velocity Coupling

For incompressible fluid flows, there are no equations to evolve the pressure. The pressure and velocity at the new time step are found by solving the pressure-velocity coupling, using the Pressure Implicit with Splitting Operators (PISO) solver, as proposed by Issa [148]. Let the momentum equation be denoted in the form of

$$a_P \mathbf{u}_P = \mathbf{H}(\mathbf{u}) - \nabla p. \tag{1}$$

The operator $\mathbf{H}$ accounts for the advection and source terms, and can be written as

$$\mathbf{H} = -\sum_f a_N \mathbf{u}_N + \frac{\mathbf{u}^0}{\Delta t}, \tag{2}$$

recalling that P denotes the owner cell and N the neighbour cell. ∇p is written as $\sum_f \mathbf{S}(p_0)_f$, summed over the cell faces f.

With these in mind, the PISO procedure is summarised as follows;

1. Get the velocity field approximation by solving the momentum equation. To calculate the pressure gradient, the value of pressure from the previous step is used. For the advection terms, use previous values of flux.

$$a_P \mathbf{u}_P = \sum_f \mathbf{S}(p_0)_f. \tag{3}$$

2. The face flux approximation is assembled for use in the pressure equation

$$\tilde{F}_f = \left(\frac{\mathbf{H}(\mathbf{u})}{a_P}\right)_f \cdot \mathbf{S}_f. \tag{4}$$

3. Using the approximate velocity values, the pressure equation is solved (the pressure solution step)

$$\sum_f \left[\left(\frac{1}{a_P} \right)_f (\nabla p)_f \right] \cdot \mathbf{S}_f = \sum_f \tilde{F}_f. \tag{5}$$

4. The final flux is found, correcting the approximated flux found by the pressure effect

$$F = \tilde{F}_f - \left[\left(\frac{1}{a_P} \right)_f (\nabla p)_f \right] \cdot \mathbf{S}_f. \tag{6}$$

5. The cell-centre velocity is corrected using the new values of pressure (the explicit velocity correction step)

$$.\mathbf{u}_P = \frac{\mathbf{H}(\mathbf{u})}{a_P} - \frac{1}{a_P} \nabla p \tag{7}$$

6. Return to Step 2 $nCorrector$ amount of times, of end.

The implementation in detail is available in [149].

Appendix

Semi-Lagrangian characteristic formulation for Euler equations

Euler's equations of inviscid gas dynamics are given as

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho v\\ E\end{bmatrix}+\frac{\partial}{\partial x}\begin{bmatrix}\rho v\\ \rho v^2+p\\ v(E+p)\end{bmatrix}=0 \tag{8}$$

where ρ is density, u is velocity, e is total energy, and p is pressure. $\mathbf{U}$ is a vector containing the conservative variables and $\mathbf{F}$ is a vector of the inviscid fluxes.

$$\mathbf{U}=\begin{bmatrix}\rho\\ \rho v\\ E\end{bmatrix},\mathbf{F}=\begin{bmatrix}\rho v\\ \rho v^2+p\\ v(E+p)\end{bmatrix} \tag{9}$$

Pressure p can be obtained from the equation of state for the perfect gas

$$p=(\gamma-1)(E-\frac{1}{2}\rho v^2) \tag{10}$$

where $\gamma=1.4$. Following the steps of semi-Lagrangian characteristic formulation as detailed by Ii and Xiao [91], the linearised version of the Euler equations about the primitive variables $\mathbf{W}$ are obtained by freezing the Jacobian matrix $\mathbf{A}$,

$$\frac{\partial \mathbf{W}}{\partial t}+\mathbf{A}\frac{\partial \mathbf{W}}{\partial x}=0, \tag{11}$$

where

$$\mathbf{W} = \begin{bmatrix} \rho \\ u \\ p \end{bmatrix}, \mathbf{A} = \begin{bmatrix} u & \rho & 0 \\ 0 & u & \frac{1}{\rho} \\ 0 & \rho c^2 & u \end{bmatrix} \tag{12}$$

where $c = \sqrt{\gamma p/\rho}$ is the speed of sound. $\mathbf{A}$ is diagonalised as

$$\mathbf{A} = \mathbf{R}\mathbf{\Lambda}\mathbf{L}. \tag{13}$$

Λ is the diagonal matrix of the eigenvalues where the non-zero elements are the characteristic speeds

$$\lambda_1 = u; \quad \lambda_2 = u + c; \quad \lambda_3 = u - c \tag{14}$$

and $\mathbf{L} = \mathbf{R}^{-1}$. The characteristic form of (11) is then

$$\mathbf{L}\frac{\partial \mathbf{W}}{\partial t} + \mathbf{\Lambda}\mathbf{L}\frac{\partial \mathbf{W}}{\partial x} = 0, \tag{15}$$

with

$$\mathbf{\Lambda} = \begin{bmatrix} u & 0 & 0 \\ 0 & u + c & 0 \\ 0 & 0 & u - c \end{bmatrix}, \mathbf{L} = \begin{bmatrix} 1 & 0 & -\frac{1}{c^2} \\ 0 & 1 & \frac{1}{\rho c} \\ 0 & 1c^2 & -\frac{1}{\rho c}. \end{bmatrix} \tag{16}$$

Using (16), the decoupled system for the characteristic variables are, for $\mathcal{C}_1(X_0), \lambda_1$:

$$d\rho - \frac{1}{c^2}dp = 0, \tag{17}$$

for $\mathcal{C}_2(X_0), \lambda_2$:

$$du + \frac{1}{\rho c}dp = 0, \tag{18}$$

and for $\mathcal{C}_3(X_0), \lambda_3$:

$$du - \frac{1}{\rho c}dp = 0. \tag{19}$$

The primitive variables at X_0 can then be found.

$$\rho(X_0) - \rho(X(\mathcal{C}_1)) - \frac{1}{c^2}[p(X_0) - p(X(\mathcal{C}_1))] = 0, \tag{20}$$

$$u(X_0) - u(X(\mathcal{C}_2)) + \frac{1}{\rho c}[p(X_0) - p(X(\mathcal{C}_2))] = 0, \tag{21}$$

$$u(X_0) - u(X(\mathcal{C}_3)) - \frac{1}{\rho c}[p(X_0) - p(X(\mathcal{C}_2))] = 0, \tag{22}$$

where $X(\mathcal{C}_1)$, $X(\mathcal{C}_2)$, $X(\mathcal{C}_3)$ are the points on the characteristic curves $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$.

For solving the Euler equations, the value of moments at step $n+1(t = t_{n+1} = t^n + \Delta t)$ are updated using a Runge-Kutta method. The following initial value is solved

$$\frac{\partial X}{\partial t} = -\lambda_m(X, t), \quad X_0 = x_{i-1/2}, \tag{23}$$

for the cell boundary point $x_{i-1/2}$, where $m = 1, 2, 3$. The third-order TVD Runge-Kutta method has the following steps for solving (23),

$$X_1(\mathcal{C}_m) = X_0 - \lambda_m(X_0, t_0)\Delta t, \tag{24}$$

$$X_2(\mathcal{C}_m) = \frac{3}{4}X_0 + \frac{1}{4}X_1 - \frac{1}{4}u\lambda_m(X_1, t_1)\Delta t, \tag{25}$$

$$X_3(\mathcal{C}_m) = \frac{1}{3}X_0 + \frac{2}{3}X_2 - \frac{2}{3}\lambda_m(X_2, t_2)\Delta t. \tag{26}$$

Solving (20)-(22) for ρ, u, p along characteristic curves, we find the point values of the variables at each R-K substep as follows,

$$\overline{{}^p p}_{i-1/2}^{<l>} = \frac{1}{2}\Big\{\mathcal{P}(X_l(\mathcal{C}_2)) + \mathcal{P}(X_l(\mathcal{C}_3)) + \overline{{}^p\rho}_{i-1/2}^{<l-1>}\overline{{}^p c}_{i-1/2}^{<l-1>}\{\mathcal{U}(X_l(\mathcal{C}_2)) - \mathcal{U}(X_l(\mathcal{C}_3))\}\Big\}, \tag{27}$$

$$\overline{{}^p u}_{i-1/2}^{<l>} = \frac{1}{2}\Big\{\mathcal{U}(X_l(\mathcal{C}_2)) + \mathcal{U}(X_l(\mathcal{C}_3)) + \frac{1}{\overline{{}^p p}_{i-1/2}^{<l-1>}\overline{{}^p c}_{i-1/2}^{<l-1>}}\{\mathcal{P}(X_l(\mathcal{C}_2)) - \mathcal{P}(X_l(\mathcal{C}_3))\}\Big\}, \tag{28}$$

$$\overline{{}^p\rho}_{i-1/2}^{<l>} = \mathcal{R}(X_l(\mathcal{C}_1)) + \frac{1}{(\overline{{}^p c}_{i-1/2}^{<l-1>})^2}\{\overline{{}^p p}_{i-1/2}^{<l-1>} - \mathcal{P}(X_l(\mathcal{C}_1))\}, \tag{29}$$

where $\mathcal{R}(x)$, $\mathcal{U}(x)$, and $\mathbb{P}(x)$ represent the CIP-CSL reconstructions of ρ, u, and p respectively. $\mathcal{R}(X_l(\mathcal{C}_m)$, $\mathcal{U}(X_l(\mathcal{C}_m)$, and $\mathbb{P}(X_l(\mathcal{C}_m)$ then denote the semi-Lagrangian solutions along the trajectories carved by the characteristic curves $\mathcal{C}_m$ for each R-K substep $l = 1, 2, 3$.

Recalling that the characteristic velocities $\lambda_m(X_0, t_l)$ are a function of primitive variable u and the speed of sound c as in Eq. 14, and recalling that $c = \sqrt{\frac{\gamma p}{\rho}}$, (27) - (29) are used to calculate λ_m at each R-K substep ($l = 0, 1, 2$).

The point values of the primitives at $x_{i-1/2}$ are updated to $n+1$ by

$$\overline{{}^{p}p}_{i-1/2}^{n+1} = \overline{{}^{p}p}_{i-1/2}^{<3>}, \tag{30}$$

$$\overline{{}^{p}u}_{i-1/2}^{n+1} = \overline{{}^{p}u}_{i-1/2}^{<3>}, \tag{31}$$

$$\overline{{}^{p}\rho}_{i-1/2}^{n+1} = \overline{{}^{p}\rho}_{i-1/2}^{<3>}. \tag{32}$$

The cell average values of the conservative variables $\mathbf{U}$ at cell i are updated as

$$\overline{{}^{V}\mathbf{U}}_{i-1/2}^{n+1} = \overline{{}^{V}\mathbf{U}}_{i-1/2}^{n} - \frac{\Delta t}{\Delta x_i}(\hat{\mathcal{F}}_{i+1/2} - \hat{\mathcal{F}}_{i-1/2}). \tag{33}$$

The numerical fluxes are found using the point values of the primitives at each R-K substep;

$$\widehat{\mathcal{F}}_{i-1/2} = \frac{\mathcal{F}_{i-1/2}^{<0>}(\overline{{}^{P}\mathbf{W}}_{i-1/2}^{<0>}) + \mathcal{F}_{i-1/2}^{<1>}(\overline{{}^{P}\mathbf{W}}_{i-1/2}^{<1>}) + 4\mathcal{F}_{i-1/2}^{<2>}(\overline{{}^{P}\mathbf{W}}_{i-1/2}^{<2>})}{6}. \tag{34}$$

The following relations of continuous physical variables are assumed for point values and cell averages

$$\overline{{}^{V}u} = \frac{\overline{{}^{V}\rho u}}{\overline{{}^{V}\rho}}; \quad \overline{{}^{V}p} = (\overline{{}^{V}e} - \frac{\overline{{}^{V}\rho u}^{2}}{2})(\gamma - 1); \tag{35}$$

Bibliography

[1] P. Rantanen, "Helsinki university of technology internal combustion engine laboratory," 2017. Accessed: 2018-12-22.

[2] J. Roenby, H. Bredmose, and H. Jasak, "A computational method for sharp interface advection," *Royal Society Open Science*, 2016.

[3] T. Yabe, R. Tanaka, T. Nakamura, and F. Xiao, "An exactly conservative semi-lagrangian scheme (CIP-CSL) in one dimension," *Monthly Weather Review*, vol. 129, no. 2, pp. 332–344, 2001.

[4] S. Koshizuka and Y. Oka, "Moving-particle semi-implicit method for fragmentation of incompressible fluid," *Nuclear Science and Engineering*, vol. 123, no. 3, pp. 421 – 434, 1996.

[5] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, and L. Tobiska, "Quantitative benchmark computations of two dimensional bubble dynamics," *International Journal of Numerical Methods in Fluids*, vol. 60, no. 11, pp. 1259 – 1288, 2009.

[6] P. Tsai, S. Pacheco, C. Pirat, and D. Lohse, "Drop impact upon micro- and nanostructured superhydrophobic surfaces," *Langmuir*, vol. 25, pp. 183 – 204, 2009.

[7] K. Yokoi, D. Vadillo, J. Hinch, and I. Hutchings, "Numerical studies of the influence of the dynamic contact angle on a droplet impacting on a dry surface," *Physics of Fluids*, vol. 21, no. 7, p. 072102, 2009.

[8] G. E. Cossali, A. Conghe, and M. Marengo, "The impact of a single drop on a wetted solid surface," *Experiments in Fluids*, vol. 22, no. 6, pp. 463–472, 1997.

[9] N. Ashgriz and J. Y. Poo, "Coalescence and separation in binary collisions of liquid drops," *Journal of Fluid Mechanics*, vol. 221, pp. 183 – 204, 1990. offset droplet collision.

[10] M. Wang, M. Broumand, and M. Birouk, "Liquid jet trajectory in a subsonic gaseous cross-flow: an analysis of published correlations," *Atomization and Sprays*, vol. 26, no. 11, pp. 1083–1110, 2016.

[11] N. Ashgriz, "Atomization of a liquid jet in a crossflow," *In Proceedings of the 4th Interntional Meeting of Advances in Thermofluids, Malaysia*, 2011.

[12] C. Iyogun, M. Birouk, and N. Popplewell, "Trajectory of water jet exposed to low subsonic cross-flow," *Atomization and Sprays*, vol. 16, no. 8, pp. 303–348, 2006.

[13] E. Farvardin, M. Johnson, H. Alaee, A. Martinez, and A. Dolatabadi, "Comparative study of biodiesel and diesel jets in gaseous crossflow," *Journal of Propulsion and Power*, vol. 29, pp. 1292–1302, 2013.

[14] M. Birouk, C. Iyogun, and N. Popplewell, "Role of viscosity on trajectory of liquid jets in a cross-airflow," *Atomization and Sprays*, vol. 17, no. 3, pp. 267–287, 2007.

[15] C. D. Bolszo, V. G. McDonnell, G. A. Gomez, and G. S. Samuelsen, "Injection of water-in-oil emulsion jets into a subsonic crossflow: An experimental study," *Atomization and Sprays*, vol. 24, no. 4, pp. 303–348, 2014.

[16] J. Mazallon, Z. Dai, and G. M. Faeth, "Primari breakup of nonturbulent round liquid jets in gas crossflows," *Atomization and Sprays*, vol. 9, no. 3, pp. 291–311, 1999.

[17] K. A. Sallam, C. Aalburg, and G. M. Faeth, "Breakup of round nonturbulent liquid jets in gaseous crossflow," *AIAA Journal*, vol. 42, no. 12, pp. 2529–2540, 2004.

[18] R. Leveque, "High-resolution conservative algorithms for advection in incompressible flow," *SIAM Journal on Numerical Analysis*, vol. 33, no. 2, pp. 627 – 665, 1996.

[19] S. A. Wormald and J. M. Coupland, "On measuring 3d flow within inkjet droplet streams using a digital holographic microscope," *Journal of Modern Optics*, vol. 57, no. 9, pp. 700–708, 2009.

[20] H. K. Versteeg, G. K. Hargrave, and M. Kirby, "Internal flow and near-orifice spray visualisations of a model pharmaceutical pressurised metered dose inhaler," *Journal of Physics: Conference Series*, vol. 45, pp. 897–908, 2004.

[21] W. Wulff, "Computer simulation of two-phase flow in nuclear reactors," *Nuclear Engineering and Design*, vol. 141, no. 1-2, pp. 303–313, 1993.

[22] E. von Berg, W. Edelbauer, A. Alajbegovic, R. Tatschl, M. Volmajer, B. Kegl, and L. C. Ganippa, "Coupled simulations of nozzle flow, primary fuel jet breakup, and spray formation," *J. Eng. Gas Turbines Power*, vol. 127, no. 4, pp. 897–908, 2004.

[23] R. R. P. Release, "Rolls-royce delivers first epa tier 4-compliant propulsion from mtu for the san francisco water emergency transportation authority," 2018. Accessed: 2019-08-27.

[24] L. University, "Rolls Royce UTC in Combustion System Aerothermal Processes," 2018. Accessed: 2019-08-27.

[25] B. Belcher, "Rolls-royce CLEEN II Low Emission Combustion Technology," 2019. Accessed: 2019-08-27.

[26] U. of Brighton, "Advanced Engineering Centre -Droplets and Sprays," 2019. Accessed: 2019-08-27.

[27] M. Ishii and T. Hibiki, *Thermo-Fluid Dynamics of Two-Phase Flow*, vol. 1. Springer, 2011.

[28] A. Jafari and N. Ashgriz, *Numerical Techniques for Free Surface Flows: Interface Capturing and Interface Tracking*, pp. 1–27. Boston, MA: Springer US, 2013.

[29] J. U. Brackbill and H. M. Ruppel, "Flip: A low-dissipation, particle-in-cell method for fluid flow," *Computer Physics Communications*, vol. 48, no. 1, pp. 25 – 38, 1988.

[30] M. J. Turner, R. W. Clough, H. C. Martin, and L. J. Topp, "3d simulation of fluid-particle interactions with the number of particles reaching 100," *Computer Methods in Applied Mechanics and Engineering*, vol. 145, no. 3-4, pp. 301 – 321, 1997.

[31] S. Hieber and P. Koumoutsakos, "A lagrangian particle level set method," *Journal of Computational Physics*, vol. 210, no. 1, pp. 342 – 367, 2005.

[32] J. Walther, T. Werder, R. Jaffe, and P. Koumoutsakos, "Hydrodynamic properties of carbon nanotubes," *Physical Review E*, vol. 69, p. 062201, 2004.

[33] E. Shams and S. Apte, "A hybrid lagrangian-eulerian approach for two-phase flows with fully resolved interfaces," *ILASS Americas 20th Annual Conference on Liquid Atomization and Spray Systems, Chicago, IL, May 2007*, 2007.

[34] G. Ryskin and L. Leal, "Numerical solution of free-boundary problems in fluid mechanics. part 1.," *Journal of Fluid Mechanics*, vol. 148, pp. 1–17, 1984.

[35] H. W. Zheng, C. Shu, and Y. T. Chew, "Lattice boltzmann interface capturing method for incompressible flows," *Physical Review E*, vol. 72, p. 056705, 2005.

[36] "Lattice-boltzmann simulations of particle-fluid suspensions," *Journal of Statistical Physics*, vol. 104.

[37] F. Harlow and J. Welch, "Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface," *Physics of Fluids*, vol. 8, pp. 2182–2189, 1965.

[38] C. W. Hirt and B. D. Nichols, "Volume of fluid (vof) method for the dynamics of free boundaries," *Journal of Computational Physics*, vol. 39, pp. 201 – 225, 1981.

[39] R. Scardovelli and S. Zaleski, "Direct numerical simulation of free-surface and interfacial flow," *Annual Review of Fluid Mechanics*, vol. 31, pp. 567 – 603, 1999.

[40] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations," *Journal of computational physics.*, vol. 79, no. 1, pp. 12–49, 1988.

[41] S. Osher and R. Fedkiw, *Level set methods and dynamic implicit surfaces*. Springer, 2003.

[42] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell, "A hybrid particle level set method for improved interface capturing," *Journal of Computational Physics*, vol. 183, pp. 83 – 116, 2002.

[43] T. Menard, "Coupling level set/vof/ghost fluid methods: Validation and application to 3d simulation of the primary break-up of a liquid jet," *International Journal of Multiphase Flow*, vol. 33, no. 5, pp. 510 – 524, 2007.

[44] G. Son and N. Hur, "A coupled level set and volume-of-fluid method for the buoyancy-driven motion of fluid particles," *Numerical Heat Transfer, Part B: Fundamentals (An International Journal of Computation and Methodology)*, vol. 42, no. 6, pp. 523 – 542, 2002.

[45] 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*, vol. 187, no. 1, pp. 110 – 136, 2003.

[46] M. Griebel and M. Klitz, "Clsvof as a fast and mass-conserving extension of the level-set method for the simulation of two-phase flow problems," *Numerical Heat Transfer, Part B: Fundamentals (An International Journal of Computation and Methodology)*, vol. 71, no. 1, pp. 1 – 36, 2016.

[47] M. Sussman, P. Smereka, and S. Osher, "A level set approach for computing solutions to incompressible two-phase flow," *Journal of Computational Physics*, vol. 114, no. 1, pp. 146 – 159, 1994.

[48] W. Noh and P. Woodward, "Slic (simple line interface calculations)," *Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics*, vol. 59, pp. 330 – 340, 1976.

[49] OpenFOAM, "OpenFOAM.org," 2018. Accessed: 2018-12-22.

[50] ANSYS, "VOF model theory," 2009. Accessed: 2018-12-31.

[51] S. Popinet, "Gerris flow solver," 2013. Accessed: 2018-12-31.

[52] Flow3D, "VOF: What is in a name," 2018. Accessed: 2018-12-31.

[53] B. D. Nichols and C. W. Hirt, "Methods for calculating multidimensional, transient free surface flows past bodies," *International Conference on Numerical Ship Hydrodynamics, Gaithersburg, Los Alamos National Laboratory, 1975*, 1975.

[54] B. Lafaurie, C. Nardone, R. Scardovell, S. Zaleski, and G. Zanetti, "Modeling merging and fragmentation in multiphase flows with surfer," *Journal of Computational Physics*, vol. 113, pp. 134 – 147, 1994.

[55] C. Ubink and R. I. Issa, "A method for capturing sharp fluid interfaces on arbitrary meshes," *Journal of Computational Physics*, vol. 153, pp. 26 – 50, 1999.

[56] N. Ashgriz and J. Y. Poo, "Flair: Flux line-segment model for advection and interface reconstruction," *Journal of Computational Physics*, vol. 93, pp. 449–468, 4 1991.

[57] D. Youngs, *Time-Dependent Multi-material Flow with Large Fluid Distortion*, ch. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws, pp. 273–285. Academic Press, 1984.

[58] D. Gueyffier, J. Li, A. Nadim, S. Scardovelli, and S. Zaleski, "Volume of fluid interface tracking with smoothed surface stress methods for three-dimensional flows," *Journal of Computational Physics*, vol. 152, no. 2, pp. 423 – 456, 1999.

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

[60] J. Brackbill, D. Kothe, and C. Zemach, "A continuum method for modeling surface tension," *Journal of Computational Physics*, vol. 100, no. 2, pp. 335 – 354, 1992.

[61] H. G. Weller, "The development of a new flame area combustion model using conditional averaging," Tech. Rep. TF/9307, Department of Mechanical Engineering, Imperial College, March 1993.

[62] E. Berberović, N. P. van Hinsberg, S. Jakirlić, I. V. Roisman, and C. Tropea, "Drop impact onto a liquid layer of finite thickness: Dynamics of the cavity evolution," *Phys. Rev. E*, vol. 79, p. 036306, Mar 2009.

[63] S. S. Deshpande, L. Anumolu, and M. F. Trujillo, "Evaluating the performance of the two-phase flow solver interfoam," *Computational Science and Discovery*, vol. 5, no. 1, pp. 014–016, 2012.

[64] S. G, "Efficient implementation of a coupled level-set and volume-of-fluid method for three-dimensional incompressible two-phase flows," *Numerical Heat Transfer, Part B: Fundamentals*, vol. 43, no. 6, pp. 549–565, 2003.

[65] Z. Wang, J. Yang, B. Koo, and F. Stern, "A coupled level set and volume-of-fluid method for sharp interface simulation of plunging breaking waves," *International Journal of Multiphase Flow*, vol. 35, no. 3, pp. 227 – 246, 2009.

[66] M. Sussman, E. Fatemi, P. Smereka, and S. Osher, "An improved level set method for incompressible two-phase flows," *Computers and Fluids*, vol. 27, no. 5-6, pp. 663 – 680, 1998.

[67] A. Albadawi, D. B. Donoghue, A. J. Robinson, D. B. Murray, and Y. M. C. Delaure, "Influence of surface tension implementation in volume of fluid and coupled volume of fluid with level set methods for bubble growth and detachment," *International Journal of Multiphase Flow*, vol. 53, pp. 11 – 28, 2013.

[68] C. Kunkelmann and P. Stephan, "Modification and extension of a standard volume-of-fluid solver for simulating boiling heat transfer," *In Proceedings of the V European Conference on Computational Fluid Dynamics (ECCOMAS CFD2010)*, 2010.

[69] M.Skarysz, A.Garmory, and M.Dianat, "A coupled level set and volume of fluid method for automotive exterior water management applications," *Journal of Computational Physics*, vol. 368, pp. 254 – 276, 2018.

[70] A. H.T and S. M., "Geometric algorithms for 3d interface reconstruction," *Proceedings of the 16th International Meshing Roundtable*, pp. 405 – 422, 2008.

[71] P. D. Lax, "Weak solutions of nonlinear hyperbolic equations and their numerical computation," *Communications on Pure and Applied Mathematics*, vol. 7, no. 1, pp. 159–193, 1954.

[72] P. Lax and B. Wendroff, "Systems of conservation laws," *Communications on Pure and Applied Mathematics*, vol. 13, no. 2, pp. 217–237, 1960.

[73] S. Godunov, "A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations," *Math. Sbornik*, vol. 47, pp. 271 – 306, 1959.

[74] S. Patankar, *Numerical Heat Transfer and Fluid Flow*. Taylor Francis, 1 ed., 1980.

[75] M. J. Turner, R. W. Clough, H. C. Martin, and L. J. Topp, "Stiffness and deflection analysis of complex structures," *Journal of the Aeronautical Sciences*, vol. 23, no. 9, pp. 805 – 823, 1956.

[76] P. W. McDonald, "The computation of transonic flow through two-dimensional gas turbine cascades," *ASME Paper*, vol. 71-GT-89, 1971.

[77] R. W. MacCormack and A. J. Paullay, "Computational efficiency achieved by time splitting of finite difference operators," *AIAA Paper*, pp. 72–154, 1972.

[78] J. Blazek, *Computational Fluid Dynamics: Principles and Applications*. Elsevier, 3 ed., 2015.

[79] Y. Imai and T. Aoki, "Accuracy study of the IDO scheme by fourier analysis," *Journal of Computational Physics*, vol. 217, no. 2, pp. 453 – 472, 2006.

[80] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, "Uniformly high order accurate essentially non-oscillatory schemes, III," *Journal of Computational Physics*, vol. 71, no. 2, pp. 231 – 303, 1987.

[81] J. M. Hyman, "Accurate monotonicity preserving cubic interpolation," *SIAM Journal on Scientific and Statistical Computing*, vol. 4, no. 4, pp. 645–654, 1983.

[82] G.-S. Jiang and C.-W. Shu, "Efficient implementation of weighted ENO schemes," *Journal of Computational Physics*, vol. 126, no. 1, pp. 202 – 228, 1996.

[83] X.-D. Liu, S. Osher, and T. Chan, "Weighted essentially non-oscillatory schemes," *Journal of Computational Physics*, vol. 115, no. 1, pp. 200 – 212, 1994.

[84] H. Takewaki, A. Nishiguchi, and T. Yabe, "Cubic interpolated pseudo-particle method (cip) for solving hyperbolic-type equations," *Journal of Computational Physics*, vol. 61, no. 2, pp. 261 – 268, 1985.

[85] R. Tanaka, T. Nakamura, and T. Yabe, "Constructing exactly conservative scheme in a non-conservative form," *Computer Physics Communications*, vol. 126, no. 3, pp. 232 – 243, 2000.

[86] S. J. Lee, I. K. Park, and J. J. Jeong, "Modified cip-csl/fv method for incompressible flows," *Computers and Fluids*, vol. 86, pp. 240 – 250, 2013.

[87] T. Nakamura, R. Tanaka, T. Yabe, and K. Takizawa, "Exactly conservative semi-lagrangian scheme for multi-dimensional hyperbolic equations with directional splitting technique," *Journal of Computational Physics*, vol. 174, pp. 171 – 207, 2001.

[88] F. Xiao, T. Yabe, X. Peng, and H. Kobayashi, "Conservative and oscillation-less atmospheric transport schemes based on rational functions," *Journal of Geophysical Research: Atmospheres*, vol. 107, no. D22, pp. ACL 2–1–ACL 2–11, 2002. 4609.

[89] F. Xiao and T. Yabe, "Completely conservative and oscillationless semi-lagrangian schemes for advection transportation," *Journal of Computational Physics*, vol. 170, no. 2, pp. 498 – 522, 2001.

[90] C.-W. Shu, "Total-variation-diminishing time discretizations," *SIAM Journal on Scientific and Statistical Computing*, vol. 9, no. 6, pp. 1073–1084, 1988.

[91] S. Ii and F. Xiao, "CIP/multi-moment finite volume method for euler equations: A semi-lagrangian characteristic formulation," *Journal of Computational Physics*, vol. 222, no. 2, pp. 849 – 871, 2007.

[92] N. Onodera, T. Aoki, and K. Yokoi, "A fully conservative high-order upwind multi-moment method using moments in both upwind and downwind cells," *International Journal for Numerical Methods in Fluids*, 2016. fld.4228.

[93] Q. Li, S. Omar, X. Deng, and K. Yokoi, "Constrained interpolation profile conservativesemi-lagrangian scheme based on third-order polynomial functions and essentially non-oscillatory (CIP-CSL3ENO) scheme," *Commun. Comput. Phys*, vol. 22, no. 3, pp. 765 – 788, 2017.

[94] C.-W. Shu and S. Osher, "Efficient implementation of essentially non-oscillatory shock-capturing schemes," *Journal of Computational Physics*, vol. 77, no. 2, pp. 439 – 471, 1988.

[95] C.-W. Shu and S. Osher, "Efficient implementation of essentially non-oscillatory shock-capturing schemes, II," *Journal of Computational Physics*, vol. 83, no. 1, pp. 32 – 78, 1989.

[96] S. Serna and A. Marquina, "Power ENO methods: a fifth-order accurate weighted power ENO method," *Journal of Computational Physics*, vol. 194, no. 2, pp. 632 – 658, 2004.

[97] J. Guo and J.-H. Jung, "Total-variation-diminishing time discretizations," *Journal of Scientific Computing*, vol. 70, no. 2, pp. 551–575, 2017.

[98] J.-M. Qiu and C.-W. Shu, "Conservative high order semi-lagrangian finite difference WENO methods for advection in incompressible flow," *Journal of Computational Physics*, vol. 230, no. 4, pp. 863 – 889, 2011.

[99] G. Hu, R. Li, and T. Tang, "A robust WENO type finite volume solver for steady euler equations on unstructured grids," *Communications in Computational Physics*, vol. 9, no. 3, pp. 627 – 648, 2011.

[100] Z. Sun, H. Teng, and F. Xiao, "A slope constrained 4th order multi-moment finite volume method with weno limiter," *Communications in Computational Physics*, vol. 18, pp. 901–930, 2015.

[101] C.-S. Huang, F. Xiao, and T. Arbogast, "Fifth order multi-moment weno schemes for hyperbolic conservation laws," *Journal of Scientific Computing*, vol. 64, no. 2, pp. 477–507, 2015.

[102] S. Zhang and C.-W. Shu, "A new smoothness indicator for the (WENO) schemes and its effect on the convergence to steady state solutions," *Journal of Scientific Computing*, vol. 31, no. 1-2, pp. 273–305, 2007.

[103] J. Qiu and C.-W. Shu, "Hermite (WENO) schemes and their application as limiters for runge-kutta discontinuous galerkin method: one-dimensional case," *Journal of Computational Physics*, vol. 193, no. 1, pp. 115–135, 2004.

[104] G. A. Sod, "A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws," *Journal of Computational Physics*, vol. 27, no. 1, pp. 1 – 31, 1978.

[105] 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*, vol. 162, no. 2, pp. 301 – 337, 2000.

[106] T. Maric, H. Marschall, and D. Bothe, "voffoam: A geometrical volume of fluid algorithm on arbitrary unstructured meshes with local dynamic adaptive mesh refinement using openfoam," *arXiv*, 2013.

[107] T. Maric, H. Marschall, and D. Bothe, "lentfoam: A hybrid level set/front tracking method on unstructured meshes," *Computers and Fluids*, vol. 113, pp. 20 – 31, 2013.

[108] K. Yokoi, "A numerical method for free-surface flows and its application to droplet impact on a thin liquid layer," *Journal of Scientific Computing*, vol. 35, no. 2, pp. 372–396, 2008.

[109] Z. Wang, J. Yang, and F. Stern, "A new volume-of-fluid method with a constructed distance function on general structured grids," *Journal of Computational Physics*, vol. 231, no. 9, pp. 3703 – 3722, 2012.

[110] D. H. Eberly, *3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics*. CRC Press, 2006.

[111] S. T. Zalesak, "Fully multidimensional flux-corrected transport algorithms for fluids," *Journal of Computational Physics*, vol. 31, no. 3, pp. 335 – 362, 1979.

[112] M. Rudman, "Volume tracking methods for interfacial flow calculations," *International Journal of Numerical Methods in Fluids*, vol. 24, no. 7, pp. 671–691, 1997.

[113] W. J. Rider and D. B. Kothe, "Reconstructing volume tracking," *Journal of Computational Physics*, vol. 141, no. 2, pp. 112 – 152, 1998.

[114] R. Forch, H. Schonherr, A. Tobias, and A. Jenkins, *Surface Design: Applications in Bioscience and Nanotechnology*, vol. 1. Wiley, 2009.

[115] C. Kunkelmann, *Numerical Modeling and Investigation of Boiling Phenomena*. PhD thesis, Technische Universitat Darmstadt, 8 2011. contact angle.

[116] L. Tanner, "The spreading of silicon oil drops on horizontal surfaces," *Journal of Physics D: Applied Physics*, vol. 12, no. 9, p. 1473, 1979.

[117] T. Abadie, J. Aubin, and D. Legendre, "On the combined effects of surface tension force calculation and interface advection on spurious currents within volume of fluid and level set frameworks," *Journal of Computational Physics*, vol. 297, pp. 611 – 636, 2015.

[118] L. Rayleigh, "On the instability of jets," *Proceedings of the London Mathematical Society*, vol. s1-10, no. 1, pp. 4 – 13, 1878. jet experiment.

[119] J. Glimm, J. W. Grove, X. L. Li, W. Oh, and D. H. Sharp, "A critical analysis of rayleigh taylor growth rates," *Journal of Computational Physics*, vol. 169, pp. 652 – 677, 2001.

[120] D. H. Sharp, "Overview of rayleigh-taylor instability," *Physica D Nonlinear Phenomena*, vol. 12, pp. 3 – 10, 1984.

[121] G. Tryggvason, "Numerical simulations of the rayleigh-taylor instability," *Journal of Computational Physics*, vol. 75, no. 2, pp. 253 – 582, 1988.

[122] J. L. Guermond and L. Quartapelle, "Isogeometric analysis of the navier-stokes-cahn-hilliard equations with application to incompressible two-phase flows," *Journal of Computational Physics*, vol. 165, no. 1, pp. 167 – 188, 2000. r-t.

[123] B. S. Hosseini, S. Turek, M. Mollier, and C. Palmes, "Isogeometric analysis of the navier-stokes-cahn-hilliard equations with application to incompressible two-phase flows," *Journal of Computational Physics*, vol. 348, no. 2, pp. 171 – 194, 2017. r-t.

[124] N. Z. Mehdizadeh, S. Chandra, and J. Mostaghimi, "Formation of fingers around the edges of a droplet hitting a metal plate with high velocity," *Journal of Fluid Mechanics*, vol. 510, pp. 353 – 373, 2004. droplet splash.

[125] K. Yokoi, "Numerical studies of droplet splashing on a dry surface: triggering a splash with the dynamic contact angle," *Soft Matter*, vol. 7, pp. 5120–5123, 2011.

[126] S. K. Betyaev, "Hydrodynamics problems and paradoxes," *Physics-Uspekhi*, vol. 38, no. 3, p. 287, 1995. splash.

[127] R. Krechetnikov and G. M. Homsy, "Crown-forming instability phenomena in the drop splash problem," *Journal of Colloid and Interface Science*, vol. 331, no. 2, pp. 555–559, 2009. droplet splash.

[128] D. A. Burzynski and S. E. Bansmer, "Droplet splashing on thin moving films at high weber numbers," *International Journal of Multiphase Flows*, vol. 101, no. 2, pp. 202–211, 2018. droplet splash.

[129] G. Liang, T. Zhang, H. Chen, H. Yu, and S. Shen, "Successive impact of multiple droplets on liquid film," *European Journal of Mechanics - B/Fluids*, 2018. droplet splash.

[130] J. W. S. L. Rayleigh, *The Theory of Sound*, vol. 2. Dover, 1896.

[131] R. Gunn, "Collision characteristics of freely falling water drops," *Science*, vol. 150, no. 3697, pp. 695–701, 1965.

[132] S. Rana and M. Herrmann, "Primary atomization of a liquid jet in crossflow," *Physics of Fluids*, vol. 23, no. 9, pp. 91–109, 2011.

[133] M. Broumand and M. Birouk, "Liquid jet in a subsonic gaseous crossflow: Recent progress and remaining challenges," *Progress in Energy and Combustion Science*, vol. 57, pp. 1–29, 2016.

[134] M. G. Pai, O. D. Desjardins, and H. Pitsch, "Detailed simulations of primary breakup of turbulent liquid jets in crossflow," *Centre for Turbulence Research Annual Research Briefs 2008*, pp. 451–465, 2008.

[135] X. Li and M. C. Soteriou, "High-fidelity simulation of high density-ratio liquid jet atomization in crossflow with experimental validation," 2014. liquid jet.

[136] J. E. Broadwell and R. E. Breidenthal, "Structure and mixing of a transverse jet in incompressible flow," *Journal of Fluid Mechanics*, vol. 148, pp. 405–412, 1984.

[137] S. Muppidi and K. Mahesh, "Study of trajectories of jets in crossflow using direct numerical simulations," *Journal of Fluid Mechanics*, vol. 530, pp. 81–100, 2005.

[138] P. K. Wu, K. A. Kirkendall, R. P. Fuller, and A. S. Nejad, "Breakup processes of liquid jets in subsonic crossflows," *Journal of Propulsion and Power*, vol. 14, pp. 173–182, 1998.

[139] C. K. Lin, J. P. Kennedy, and T. A. Jackson, "A review on penetration heights of transverse liquid jets in high-speed flows," *15th annual conference on liquid atomization and spray systems*, pp. 345–349, 2003.

[140] J. Stenzler, J. Lee, and D. Santavicca, "Penetration of liquid jets in a crossflow," *41st aerospace sciences meeting and exhibit, AIAA*, 2003.

[141] O. Elshamy and S. M. Jeng, "Study of liquid jet in crossflow at elevated ambient pressures," *ILASS-Americas*, 2005.

[142] S. Tambe, S. Jeng, H. Mongia, and G. Hsiao, "Liquid jets in subsonic crossflow," *43rd AIAA aerospace sciences meeting and exhibit*, 2005.

[143] K. Ahn, J. Kim, and Y. Yoon, "Effects of orifice internal flow on transverse injection into subsonic crossflows: cavitation and hydraulic flip," *Atomization and Sprays*, vol. 16, no. 1, pp. 1083–1110, 2006.

[144] A. Bellofiore, A. Cavaliere, and R. Ragucci, "Air density effect on the atomization of liquid jets in crossflow," *Combustion Science and Technology*, vol. 179, no. 1-2, pp. 319–342, 2007.

[145] R. Ragucci, A. Bellofiore, and A. Cavaliere, "Trajectory and momentum coherence breakdown of a liquid jet in high-density air cross-flow," *Atomization and Sprays*, vol. 17, no. 1, pp. 47–70, 2007.

[146] Q. Wang, U. Mondragon, C. Brown, and V. McDonell, "Characterization of trajectory, break point, and break point dynamics of a plain liquid jet in a crossflow," *Atomization and Sprays*, vol. 21, no. 3, pp. 203–219, 2011.

[147] K. Ahn, J. Kim, and Y. Yoon, "Correlations for penetration height of single and double liquid jets in cross flow under high-temperature conditions," *Atomization and Sprays*, vol. 21, no. 8, pp. 673–686, 2011.

[148] R. Issa, "Solution of the implicitly discretised fluid flow equations by operator-splitting," *Journal of Computational Physics*, vol. 62, pp. 40 – 65, 1986.

[149] S. M. Damian, *An Extended Mixture Model for the Simultaneous Treatment of Short and Long Scale Interfaces*. PhD thesis, National University of the Littoral, 2013.