

Unstructured finite volume methods for two-phase flows with high density ratios

Unstrukturierte Finite-Volumen-Methoden für Zweiphasenströmungen mit hohen Dichteverhältnissen

Zur Erlangung des Grades eines Doktors der Naturwissenschaften (Dr. rer. nat.)

Genehmigte Dissertation von Jun Liu aus Xiangyang, China

Tag der Einreichung: 12.July 2024, Tag der Prüfung: 30.September 2024

Erstreferent: Dr.-Ing. Tomislav Marić

Koreferent: Prof. Dr. rer. nat. Dieter Bothe

Darmstadt, Technische Universität Darmstadt



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Mathematics Department



Mathematical
Modeling and Analysis

Lagrangian / Eulerian
numerical methods for
multiphase flows

Unstructured finite volume methods for two-phase flows with high density ratios
Unstrukturierte Finite-Volumen-Methoden für Zweiphasenströmungen mit hohen Dichteverhältnissen

Accepted doctoral thesis by Jun Liu

Date of submission: 12.July 2024

Date of thesis defense: 30.September 2024

Darmstadt, Technische Universität Darmstadt

Bitte zitieren Sie dieses Dokument als:

URN: urn:nbn:de:tuda-tuprints-282663

URL: <http://tuprints.ulb.tu-darmstadt.de/28266>

Jahr der Veröffentlichung auf TUprints: 2024

Dieses Dokument wird bereitgestellt von tuprints,

E-Publishing-Service der TU Darmstadt

<http://tuprints.ulb.tu-darmstadt.de>

tuprints@ulb.tu-darmstadt.de

Die Veröffentlichung steht unter folgender Creative Commons Lizenz:

Namensnennung 4.0 International

<https://creativecommons.org/licenses/by/4.0/>

This work is licensed under a Creative Commons License:

Attribution 4.0 International

<https://creativecommons.org/licenses/by/4.0/>

For my parents

刘太德, 吕正英

Acknowledgements

First and foremost, I extend my heartfelt thanks to my first supervisor Dr.-Ing. Tomislav Marić, who has been a guiding light since my days as his master's student in 2020. Under his persistent mentorship, I have evolved from a freshman in OpenFOAM to an eligible PhD candidate. His patience, expertise, and continuous support have been instrumental in shaping my academic and personal growth.

I would also like to express my sincere appreciation to Prof. Dr. rer. nat. Dieter Bothe, whose invaluable guidance, continuous support, and insightful feedback are the cornerstone of my research journey. His expertise and encouragement have been essential in the completion of this dissertation. My sincere thanks also go to my committee members, Prof. Dr. Jan Giesselmann and Prof. Dr. Ulrich Reif, of my dissertation. Your insightful comments and corrections have significantly enhanced the quality and clarity of my work, ensuring that it meets the highest academic standards.

I am fortunate to have worked alongside colleagues in MMA group and BOSCH group, especially Guoliang, Hassan, Lisanne, Moritz, Suraj, Huijie, Luise and Tobi. Our time together has been filled with enlightening discussions that have broadened my perspectives and deepened my understanding. Equally important have been the fun times we have shared, which have made this journey enjoyable and memorable.

To my parents in Chinese: 对于我的父母刘太德，吕正英，我要表达我最特别的感谢。对我在德国读研你们给与了最坚定的情感与经济支持，这是我能在这里完成博士学业的前提。尽管你们受到教育有限，但你们始终相信教育的价值，并无条件地支持我对学历和学术的追求，你们的付出和对我的信念一直是我前进的最重要动力来源。

Lastly, I am profoundly thankful to the friends I made in Germany, particularly *Dr. Yanqin Fu*, Dr. Bingxiang Wang, Dengyi Wang, Ru Fang, Prof. Qingwen Dai, Pencheng Hu, Jinxue

Ding and Dr. Wei Li and so on. Your companionship transformed my dull off-work hours into moments of joy and relaxation, bringing balance to my life during the demanding periods of my PhD program.

This dissertation is not only a reflection of my hard work but also a testament to the support, encouragement, and love I have received from each one of you. Thank you for being part of my academic journey.

Abstract

The dissertation presents a comprehensive study of advanced computational methods for simulating incompressible two-phase flows, particularly addressing challenges associated with high-density ratios, mass and momentum conservation, and non-orthogonality errors in unstructured Finite Volume methods. The thesis extends the unstructured Level Set / Front Tracking (LENT) method, introducing the ρ LENT approach to ensure numerical consistency between mass and momentum conservation in the collocated Finite Volume discretization of the single-field two-phase Navier-Stokes equations. This method demonstrates exact numerical stability for two-phase momentum advection across a wide range of density and viscosity ratios, effectively handling challenging fluid pairings such as mercury/air and water/air, and scenarios involving strong interactions between phases.

Further, the study applies the consistency conditions derived for the ρ LENT method to geometric flux-based Volume-of-Fluid (VOF) methods. It reveals that standard computations of mass fluxes in these methods can disrupt the equivalence between scaled volume fraction equations and mass conservation equations, depending on temporal and convective term discretization schemes. The thesis proposes a dual solution approach: a consistent combination of temporal discretization and interpolation schemes, and an auxiliary mass conservation equation with a geometric calculation of face-centered densities. This approach is validated for extensive density and viscosity ratios, demonstrating its robustness and effectiveness.

Additionally, the dissertation tackles non-orthogonality errors in unstructured Finite Volume methods, which can compromise force-balanced discretization in simulating incompressible two-phase flows. A novel, deterministic residual-based control of non-orthogonality correction is introduced, removing the number of non-orthogonality corrections as a global parameter from the simulation process. This method ensures force balance, particularly for

surface tension and gravity forces, and is verified on polyhedral unstructured meshes with different non-orthogonality levels.

Overall, this dissertation provides contributions by developing, verifying and validating advanced methodologies to accurately and efficiently simulating incompressible two-phase flows under complex conditions. These developments have improved applications in industrial multiphase microfluidics, where precise computational fluid dynamics is crucial.

Zusammenfassung

Die vorliegende Dissertation präsentiert neue Entwicklungen im Bereich fortgeschrittener Berechnungsmethoden zur Simulation inkompressibler Zweiphasenströmungen, wobei insbesondere Herausforderungen im Zusammenhang mit hohen Dichteverhältnissen, Massen- und Impulserhaltung sowie Nichtorthogonalitätsfehlern in unstrukturierten Finite-Volumen-Methoden adressiert werden. Die Dissertation erweitert die unstrukturierte Level-Set / Front-Tracking (LENT) Methode durch Einführung der neuen ρ LENT Methode, mit deren Hilfe die numerische Konsistenz zwischen Masse- und Impulserhaltung in der kollabierten Finite-Volumen-Diskretisierung der einphasigen Zweiphasen-Navier-Stokes-Gleichungen gewährleistet werden kann. Diese Methode zeigt exakte numerische Stabilität für die Impulsadvektion in Zweiphasenströmungen über ein breites Spektrum an Dichte- und Viskositätsverhältnissen. Dies ermöglicht die numerische Simulation von Zweiphasenströmungen für herausfordernde Fluidpaarungen wie Quecksilber/Luft und Wasser/Luft sowie Szenarien mit starken Wechselwirkungen zwischen den Phasen.

Darüber hinaus wendet die Studie die für die ρ LENT Methode abgeleiteten Konsistenzbedingungen auf geometrische, flussbasierte Volume-of-Fluid (VOF) Methoden an. Es wird gezeigt, dass Standardberechnungen des Massenflusses in diesen Methoden die Äquivalenz zwischen skalierten Volumenfraktionsgleichungen und Massenerhaltungsgleichungen zerstören können, abhängig von den Diskretisierungsschemata für zeitliche und konvektive Terme. Die Dissertation schlägt einen dualen Lösungsansatz vor: eine konsistente Kombination aus zeitlicher Diskretisierung und Interpolationsschemata sowie eine Hilfsmassenerhaltungsgleichung mit einer geometrischen Berechnung der dichtebezogenen Flächenmitte. Dieser Ansatz wird für umfangreiche Dichte- und Viskositätsverhältnisse getestet und zeigt seine Robustheit und Effektivität.

Zusätzlich befasst sich die Dissertation mit Nichtorthogonalitätsfehlern in unstrukturierten Finite-Volumen-Methoden, welche die kraftbalancierte Diskretisierung bei der Simulation inkompressibler Zweiphasenströmungen beeinträchtigen können. Eine neue, deterministische, residual-basierte Steuerung der Nichtorthogonalitätskorrektur wird eingeführt, wobei die Anzahl der Nichtorthogonalitätskorrekturen als globaler Parameter aus dem Simulationsprozess entfernt wird. Diese Methode gewährleistet eine Balance von Kräften, insbesondere bzgl. Oberflächenspannungs- und Schwerkraftkräfte, und wird auf polyedrischen unstrukturierten Gittern mit unterschiedlichen Nichtorthogonalitätsgraden geprüft.

Insgesamt leistet diese Dissertation einen Beitrag zur hochgenauen numerischen Berechnung von inkompressibler Zweiphasenströmungen unter komplexen Bedingungen durch die Entwicklung, Verifizierung und Validierung fortschrittlicher Methoden. Diese Entwicklungen haben die Anwendungen in der industriellen Mehrphasen-Mikrofluidik verbessert, bei der eine präzise rechnergestützte Fluidodynamik entscheidend ist.

Pulications

This thesis incorporates ideas, tables, figures, and text that have previously appeared in the following publications:

- [1] **Jun Liu**, Tobias Tolle, Dieter Bothe, and Tomislav Marić. “An unstructured finite-volume Level Set / Front Tracking method for two-phase flows with large density-ratios”. In: *Journal of Computational Physics* (2023). URL: <https://doi.org/10.1016/j.jcp.2023.112426>.
- [2] **Jun Liu**, Tobias Tolle, Davide Zuzio, Jean-Luc Estivalèzes, Santiago Marquez Damian, and Tomislav Marić. “Inconsistencies in unstructured geometric volume-of-fluid methods for two-phase flows with high density ratios”. In: *Computers & Fluids* 281 (2024), p. 106375. URL: <https://doi.org/10.1016/j.compfluid.2024.106375>.
- [3] **Jun Liu**, Tobias Tolle, and Tomislav Marić. “A residual-based non-orthogonality correction for force-balanced unstructured Volume-of-Fluid methods”. *Computers and Mathematics with Applications - Under Review*, Jan 2024.
- [4] Guoliang Chai, **Jun Liu**, Dieter Bothe, Tobias Tolle, Junwei Su, and Tomislav Marić. “Curvature approximation for the unstructured geometrical Volume-of-Fluid method”. *In Preparation*, Jan 2024.

Contents

Foundations	1
1 Introduction	1
2 Mathematical model	3
2.1 Single phase fluid dynamics	3
2.2 Two-phase fluid dynamics with interface	4
3 Numerical method	8
3.1 Finite Volume Method	8
3.2 Spatial discretization	11
3.3 Equation discretization	17
3.4 Solution algorithms for pressure-velocity coupling	23
3.5 Interface capturing methods	30
Hybrid Level Set / Front Tracking method with high density-ratios	43
4 Introduction	43
5 Methodology review	52
5.1 Numerical consistency of momentum convection term	53
5.2 A semi-implicit solution algorithm for high-density ratios	58
5.3 Volume correction method	68
6 Verification and validation	72
6.1 Time step size	72
6.2 Translating droplet	73
6.3 Oscillating droplet	79
6.4 Rising bubble	83

Inconsistencies in Unstructured Geometric Volume-of-Fluid Methods for Two-Phase		
Flows with High Density Ratios		87
7	Introduction	87
8	Methodology review	92
8.1	Single-field mass conservation and volume fraction conservation	92
8.2	Collocated segregated solution algorithm with the auxiliary density equation	101
8.3	Phase-specific face area calculation	104
8.4	Consistency of Volume of Fluid (VOF) methods for Two-Phase Flows with High Density Ratios	105
9	Verification and validation	106
9.1	Time step size	106
9.2	Translating droplet in ambient flow	107
9.3	Translating sub-millimeter droplet with realistic physical properties	114
9.4	Mixing layer	119
9.5	Validation of a single rising bubble	121
9.6	Liquid jet in high speed gaseous cross-flow	122
 A residual-based non-orthogonality correction for force-balanced unstructured Volume-		
of-Fluid methods		131
10	Introduction	131
11	Methodology review	132
11.1	Solution algorithm	133
11.2	Force Balance	134
11.3	Force Balance on Non-Orthogonal Meshes	135
12	Results	144
12.1	Stationary droplet in equilibrium	144
12.2	Stationary water column in equilibrium	153
 Summary and outlook		157

Appendices	163
1 Supplementary results	163
2 Least square method	172
3 Correct cyclic boundary condition for the plicRDF-isoAdvector method	174
4 Viscous term modelling	179
5 Gravity term modelling	181

List of Figures

1	The domain Ω , split by the fluid interface $\Sigma(t)$ into two sub-domains Ω^\pm	5
2	Exemplary general mesh cell Ω_{O_f}	12
3	Four mesh quality metrics	16
4	Exemplary illustration of interface approximation using Front Tracking Method (FTM) at time t : the red dashed curve indicates the exact interface $\Sigma(t)$, the black line segments connected by marker points (black dots) construct the front $\tilde{\Sigma}(t)$ with front normal $\mathbf{n}_{\tilde{\Sigma}(t)}$ of each segment, the k th marker points $\mathbf{x}_{\tilde{\Sigma}}^k$ moves with the interpolated velocity $\tilde{\mathbf{v}}^k$	33
5	Exemplary illustration of interface approximation using Level Set Method (LSM) at time t	35
6	Exemplary illustration of interface approximation using hybrid Level Set / Front Tracking (LENT) at time t	37
7	Exemplary illustration of interface approximation using VOF at time t with marked volume fraction in each cell.	38
8	PLIC interface approximated by Youngs' algorithm [1] (blue lines) and exact interface (red circle).	40
9	A two-phase fixed control volume Ω_c separated by the interface $\Sigma(t)$	58
10	Updating the face-centered (mass flux) density in the Density-LENT (ρ LENT) method.	61
11	Computing area fractions from signed distances in the method.	64
13	The volume correction method: iso-value compensates the volume-change.	68

12	Flowchart of the ρ LENT method. The dashed blocks denote the new and modified elements of the Segregated Accuracy-driven Algorithm for Multiphase Pressure-Linked Equations (SAAMPLE) method [2]. The indices o_{max} and i_{max} in the flowchart indicate the maximal iteration numbers for the outer and inner loop, respectively, while tol_{ls} denotes the prescribed linear solver tolerance.	71
14	Translating droplet case setup.	74
15	Half section of mesh $N = 64$, droplet at initial position.	75
16	Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$: the left figure depicts the results from SAAMPLE algorithm, the right shows the results from ρ LENT method.	76
17	Comparison of the strong interface deformation with SAAMPLE method (left) and the numerically consistent interface shape of the ρ LENT method. Parameters: $N = 64$, $\rho^-/\rho^+ = 10^4$, $t = 0.0008s$	76
18	Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ for the viscous flow with surface tension forces: the left diagram depicts the results from the SAAMPLE method, and the right diagram contains the results from the ρ LENT method. The legends of these diagrams are large, and the full information is available in Appendix : fig. 49 for fig. 18a, fig. 50 for fig. 18b.	77
19	Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ with pure advection: ρ LENT method used in simulating two-phase flows with different density ratios, mesh resolution: $N \in (16, 32, 64)$	80
20	Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ with the effect of viscosity and surface tension: ρ LENT method used in simulating two-phase flows with different density ratios, mesh resolution: $N = 16, 32, 64$	81
21	L_1 norms of the oscillation frequency errors.	83
22	SAAMPLE method: the collapsed bubble shape caused by numerical inconsistency.	84
23	ρ LENT: the temporal evolution of bubble's shape with resolution $N = 128$	85

24	The average bubble velocities from ρ LENT with four resolutions are compared with: the experimental results (black solid line) from Bhaga and Weber [3], the simulation results (black dashed double-dotted line) from Hua and Lou [4], and the extracted simulation results (blue solid dotted line) from Anjos et al. [5]. Simu. and exp. are the abbreviation of simulation and experiment.	86
25	Geometric upwinding for $ V_f^\alpha _s^{isoAdvect}$ in PLIC+RDF (plicRDF)-isoAdvect [6].	99
26	Interface reconstructed as $\tilde{\Sigma}_{U,D}^o$ in upwind (U) and downwind (D) cells. Green polygons are interface polygons $\tilde{\Sigma}_{U,D}^o \cap \Omega_{U,D}$. Blue lines are intersection line segments $\tilde{\Sigma}_{U,D}^o \cap S_f$. Red points are intersection points $\tilde{\Sigma}_{U,D}^o \cap \partial S_f$	104
27	Temporal evolution of normalized mass conservation error with different schemes: interIsoFoam, $N = 64$	110
28	Temporal evolution of normalized momentum conservation error with different schemes: interIsoFoam, $N = 64$	111
29	Temporal evolution of sphericity error with different schemes: interIsoFoam, $N = 64$	112
30	Final shape of the droplet calculated by interIsoFoam with different schemes: $N = 64$	113
31	Mesh convergence study for sphericity error: interIsoFoam, Euler+upwind, $N \in (36, 48, 64, 96, 128)$	116
32	Temporal evolution of the velocity error norm $L_\infty(\mathbf{v})$ with pure advection: Euler, Gauss upwind, density ratio: 10^4 , mesh resolution: $N \in (16, 32, 64, 96, 128)$	117
33	Temporal evolution of the velocity error norm $L_\infty(\mathbf{v})$ with pure advection - combining 10 schemes, density ratio is 10^4 , mesh resolution is $N = 96$. Only Euler and upwind(ing) schemes remain consistent and stable.	118
34	2D mixing layer	119
35	Time evolution of normalized momentum error of mixing layer with different schemes and DYJEAT codes, density ratio: 10^3 , resolution: $N = 256$	120
38	Liquid in a cross flow.	123

36	Temporal evolution of rising velocity using interIsoFoam and interIsoFoam: Euler + upwind, $\rho^-/\rho^+ \approx 10^3$	126
37	Comparisons of final shapes of rising bubbles using interIsoFoam and interIsoRhoFoam with the experimental visualization from Bhaga and Weber [3] (reprinted with permission): Euler + upwind, red line from interIsoRhoFoam, blue line from interIsoFoam, $\rho^-/\rho^+ \approx 10^3$, $N = 128$, $t = 6$	127
39	The instantaneous liquid jet shape at final time $t = 7.1ms$ with different resolutions (the blue translucent jet: $N_h = [256, 128, 128]$; the gray jet: $N_l = [128, 64, 64]$) and its comparison with the experimental results(the red line). This case makes it possible to evaluate performance on coarser meshes as resolving finer structures does not impact the jet trajectory.	128
40	The shape of the injected liquid with interIsoFoam and interIsoRhoFoam (Euler and Gauss upwind, density ratio: 816, CFL number: $CFL = 0.2$, resolution: $N_l = [128, 64, 64]$), and with DYJEAT (resolution: $N = [1024, 512, 512]$ [7])	129
41	Representation of non-orthogonality: \mathbf{x}_{O_f} , \mathbf{x}_{N_f} are the centroids of two adjacent cells O , N ; \mathbf{d}_f is the vector connecting \mathbf{x}_{O_f} and \mathbf{x}_{N_f}	136
42	Three common non-orthogonality correction approaches: minimum correction (left), orthogonal correction (middle), over-relaxed correction(right). Vector $\hat{\mathbf{d}}_f$ is the unit vector of \mathbf{d}_f , i.e., $\hat{\mathbf{d}}_f := \frac{\mathbf{d}_f}{ \mathbf{d}_f }$	136
43	A sliced cell layer from a cubic computational domain with three mesh types: equidistant mesh (<i>blockMesh</i>), perturbed hexahedral mesh (<i>perturbMesh</i>) and polyhedral mesh (<i>polyMesh</i>).	145
44	The temporal evolution and average of non-orthogonal correction times, and the CPU time with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = 10$) for a stationary droplet in equilibrium on perturbMesh with different resolutions.	150
45	The temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = [1, 2, 10]$) for a stationary droplet in equilibrium on perturbMesh with different resolutions.	151
46	The velocity field of the stationary droplet in equilibrium on perturbMesh with the resolution $\frac{\Delta x}{L} = 1/60$ at $t = t_{end}$: the glyph arrows are scaled by $1e-7m/s$	152

47	The velocity field of the stationary water column in equilibrium on perturbMesh with the resolution $\frac{\Delta x}{L} = 1/60$ at $t = t_{end}$: the glyph arrows are scaled by $1e-5m/s$	156
48	Full figure of fig. 16a	163
49	Full figure of fig. 18a	164
50	Full figure of fig. 18b	165
51	Temporal evolution of normalized mass and momentum conservation error with different schemes for the case of Translating droplet in ambient flow: interIsoFoam, $N = 64$, density ratio = 1.	166
52	Temporal evolution of the velocity error norm $L_\infty(\mathbf{v})$ with pure advection - combining 10 schemes, density ratio is 1, mesh resolution is $N = 64$. All schemes are stable.	166
53	Temporal evolution of normalized mass and momentum conservation error using CrankNicolson + upwind with different resolutions for the case of Translating droplet in ambient flow: interIsoFoam, $N = 64, 96, 128$, density ratio = 10^6	167
54	The shape of the exploded injected liquid with interIsoFoam (Euler and cubic, density ratio: 816, CFL number: $CFL = 0.2$	167
55	The temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ using ResNonOrthCorr and FixNonOrthCorr ($N_{non} = 10$) with least-square gradient reconstruction for a stationary droplet in equilibrium on perturbMesh with different resolutions. These results show that there is no influence on the gradient scheme on the proposed method.	168
56	The temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ using MULES with ResNonOrthCorr for a stationary droplet in equilibrium on perturbMesh with different resolutions. These results show that the proposed method is directly applicable to the algebraic VOF method.	169

57	The temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = [1, 2, 10]$) for a stationary water column in equilibrium on perturbMesh with different resolutions. Although $N_{non} = 10$ achieves force-balance, it is a problem-dependent "free" parameter that the proposed method does not use.	170
58	The temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = [1, 2, 10]$) for a stationary column in equilibrium on polyMesh with different resolutions. Although the polyhedral mesh has very low non-orthogonality in the bulk, near walls, even for a cubic domain, non-orthogonality is substantial, and the force-imbalance is efficiently and effectively restored by the proposed method.	171
59	The ghost cells layer to correct signed distance: the green line (—) presents fixed RDF from cell centroid \mathbf{x}_c^{nei} to interface in the cyclic neighbor cell, while the red line (—) depicts the original unfixed RDF.	174
60	A droplet moves with the ambient constant flow.	177
61	The alpha field and interfaces reconstructed by iso-Alpha method reach to the right cyclic boundary; blue region: the ambient flow; red region: the liquid droplet; white line segments: the PLIC interfaces.	178
62	The alpha field and interfaces reconstructed by plic-RDF method cross the right cyclic boundary; blue region: the ambient flow; red region: the liquid droplet; white line segments: the PLIC interfaces.	178
63	A cell Ω_C is separated by the interface Σ into two regions: the shaded region Ω_C^- is occupied by the flow with density ρ^- , the rest region Ω_C^+ is occupied by another flow with density ρ^+	182

List of Tables

1	The parameters range of the case group 1 and the case group 2.	74
2	Realistic fluid properties are combined into four tests: water droplet/air ambient, mercury droplet/air ambient, silicone oil droplet/air ambient, silicone oil droplet/water ambient.	78
3	Serial execution time for the ρ LENT method.	82
4	The analysis of the oscillation frequency convergence and its comparison between the SAAMPLE method and the consistent ρ LENT method.	83
5	The combinations of different time and convection schemes used to test the effect of numerical consistency.	109
7	Realistic fluid properties of the mercury droplet/air ambient pair.	114
6	The terminated time and the final normalized errors of interIsoRhoFoam and interIsoFoam with different scheme combinations and mesh resolutions for translating droplet in ambient flow case.	115
8	Physic properties of water/air pair.	145
9	Maximum global and near-interface non-orthogonality, the velocity and pressure jump errors, and CPU times for a stationary droplet in equilibrium at the end time $t_{end} = 0.1s$	146
10	The performances of FixNonOrthCorr with non-orthogonality loop numbers larger than one for a stationary droplet in equilibrium case on perturbMesh. .	149

11	The performances of ResNonOrthCorr and FixNonOrthCorr($N_{non} = 10$) with least-square cell center gradient reconstruction method for a stationary droplet in equilibrium case on perturbMesh.	149
12	The performances of combining Multidimensional Universal Limiter for Explicit Solution (MULES) phase advection algorithm with ResNonOrthCorr for a stationary droplet in equilibrium case on perturbMesh.	152
13	The maximum non-orthogonalities in the global region and the local region near the interface, the velocity, pressure jump errors, and CPU time for a stationary water column in equilibrium at the end time $t_{end} = 0.1s$	154
14	The performances of FixNonOrthCorr with non-orthogonality loop numbers larger than one for a stationary water column in equilibrium case on perturbMesh and polyMesh.	155

Acronyms

CBC Convection-Boundedness Criterion

CDS Central Differencing Scheme

CMOM Consistent Momentum-Mass

CLSVOF Coupled Level Set and Volume Fluid Method

CBLSVOF Consistent Balanced-force Level Set and Volume Of Fluid

CFL Courant-Friedrichs-Lewy

CSF Continuum Surface Force

CRMT Consistent Rescaled Momentum Transport

CUI Cubic Upwind Interpolation

CUBISTA Convergent and Universally Bounded Interpolation Scheme for the Treatment of Advection

CV Control Volume

DNS Direct Numerical Simulations

FCT Flux Corrected Transport

FTM Front Tracking Method

FV Finite Volume

FVM Finite Volume Method

GFM Ghost Fluid Method

HOMP High-Order Momentum Preserving
LENT hybrid Level Set / Front Tracking
LFRM Local Front Reconstruction Method
LSM Level Set Method
LSF Least Squares Fit
LCRM Level Contour Reconstruction Method
LUST Linear-Upwind Stabilised Transport
MPI Message Passing Interface
MAC Marker-And-Cell
MULES Multidimensional Universal Limiter for Explicit Solution
NSE Navier-Stokes Equations
PDE Partial Differential Equation
PIMPLE PISO + SIMPLE
PISO Pressure-Implicit Split-Operator
PLIC Piecewise Linear Interface Calculation
plicRDF PLIC+RDF
QUICK Quadratic Upstream Interpolation for Convective Kinematics
RDF Reconstructed Distance Function
RK Runge-Kutta
SSP-RK3 3-order accurate Strong Stability Preserving Runge-Kutta
SAAMPLE Segregated Accuracy-driven Algorithm for Multiphase Pressure-Linked Equations
SIMPLE Semi-Implicit Method for Pressure-Linked Equations
TVD Total Variation Diminishing
THINC Tangent of Hyperbola Interface Capturing with Quadratic surface representation

THINC/QQ Tangent of Hyperbola Interface Capturing with Quadratic surface
representation and Gaussian Quadrature

UFVM Unstructured Finite Volume Method

VOF Volume of Fluid

WENO Weighted Essentially Non-Oscillatory

ρ **LENT** Density-LENT

Nomenclature

:	Double inner product	
α	Volume fraction	
χ	Phase indicator	
δ_Σ	Interface Dirac distribution	
\dot{m}	Mass flux	kg/s
\mathbf{g}	Gravitational acceleration	m/s ²
κ	Curvature	1/m
λ	Interpolation factor	
λ^p, λ^v	Under-relaxation factor for pressure and velocity	
\mathbf{b}	Body force source term in discretized momentum equation	
\mathbf{H}	Explicit velocity operator in discretized momentum equation	
\mathbf{I}	Unit tensor	
\mathbf{n}_Σ	Interface normal	
\mathbf{S}_f	Face area normal	m ²
\mathbf{x}_Σ	Center of interface	m

μ	Viscosity	m^2/s
∇_{Σ}	Interface gradient	
\otimes	Outer product	
∂_t	Partial derivative of time	
\perp	Implicit orthogonal contribution	
Σ	Interface	
σ	Surface tension coefficient	kg/s^2
θ_f	Non-orthogonal angle of cell face f	$^{\circ}$
\mathbf{x}_c	Center position of cell Ω_c	m
\mathbf{x}_f	Center position of cell face f	m
\mathbf{x}_{v_i}	Position of cell vertex v_i	m
a_c	Sum of coefficients in discretized momentum equation	
A_f	Submerged area of cell face f	m^2
d	Distance vector	m
e	Order of discretization error	
F_c	Set of faces of cell Ω_c	
F_f	Volumetric flux	m^3/s
h	Cell size	m
i	Inner iteration	
L_{∞}	Infinity error norm	

Mo	Morton number	
N_f	Index of neighbor cell of cell face f	
o	Outer iteration	
O_f	Index of owner cell of cell face f	
P	Total pressure	$\text{kg}/(\text{m s}^2)$
p	Dynamic pressure	$\text{kg}/(\text{m s}^2)$
sgn	Sign function	
T	Transpose of a tensor	
tol_{rel}/tol_{abs}	Relative/absolute tolerance	
V_f^α	Fluxed phase-specific volume	
\mathbf{f}_v	Volume forces	
\mathbf{T}	Stress tensor	$\text{kg}/(\text{m s}^2)$
Ω	Simulation domain	
Ω^+	Subset of Ω occupied by phase +	
Ω^-	Subset of Ω occupied by phase –	
Ω_c	Mesh cell / a finite volume	
ρ	Density	kg/m^3
\mathbf{v}	Velocity	m/s

Foundations

1 Introduction

Two-phase flow processes, where two distinct phases interact, are pervasive in both industrial applications and natural phenomena. These processes often involve fluids with high density ratios, such as the atomization of fuel jets [8], sloshing tanks [9], mold filling [10], water flooding [11], oil and water separation in the petroleum industry [12], and air-water interactions in environmental systems like ocean waves [13]. The investigation of two-phase flows is crucial due to their widespread occurrence and significant impact on the efficiency, safety, and environmental sustainability of various operations. For instance, understanding microfluidic systems is essential for optimizing the design of surface-tension-driven Lab-On-a-Chip components [14] and preventing fluid penetration caused by unforeseen microfractures along the sealing joints [15].

With the advancement of computational capabilities, Numerical methods for Direct Numerical Simulations (DNS) of two-phase flows have gained considerable attention [16]. These methods are vital for providing detailed insights into flow dynamics that are often impractical or impossible to obtain through experimental studies alone. In the last decades, there has been a surge in the development of new numerical techniques aimed at enhancing accuracy, reducing computational costs, and extending the capability of predictively simulating complex physical processes. This is particularly relevant for handling high density-ratio flows, which present unique challenges due to the significant differences in the properties of the interacting phases.

The geometrical complexity of flow domains in both industrial and natural processes

does not allow for exact domain discretization, which makes it very challenging to obtain a curved discretization of the domain boundary required by higher-order methods. Therefore, the second-order numerical Partial Differential Equation (PDE) discretization method, i.e., Unstructured Finite Volume Method (UFVM) is adopted in this dissertation to reduce the error significantly enough with increasing mesh resolution while ensuring a simple domain discretization workflow [17]. The flexibility of UFVM in handling irregular mesh structures makes it ideal for capturing the intricate details of the flow domain, which is often a limiting factor in structured grid approaches. Additionally, this method's local conservation properties and the ability to easily incorporate various boundary conditions make it a robust choice for a wide range of applications.

Simulating two-phase flows, especially those involving high density ratios, presents additional significant challenges. One of the most challenging tasks is accurately capturing the sharp interface between the two phases [18]. This interface represents an evolving discontinuity of physical properties such as density/viscosity and undergoes strong deformation and topological changes such as breakup and coalescence. Capturing these dynamics is complex due to the interface's inherent instability and the need to accurately resolve interfacial force jumps. Additionally, ensuring numerical stability and physical fidelity in the presence of large property variations across the interface adds to the computational challenge.

This dissertation introduces a general method to address these hurdles of robustly and accurately simulating incompressible high density ratios two-phase flows on unstructured mesh and implements the method with different interface tracking approaches. This general method, as outlined in the following chapters, has been developed on the OpenFOAM® platform [17, 19], an open-source CFD toolbox widely recognized for its modularity, robustness, and extensive user community. OpenFOAM® offers a flexible environment for customizing and extending standard solvers, making it an ideal platform for implementing and testing novel computational approaches like those proposed in this thesis.

2 Mathematical model

This section outlines the equations that govern a two-phase flow system characterized by a sharp interface between fluids or phases. Initially, the standard conservation equations for each flow are presented without considering the interface. Subsequently, the focus shifts to the mathematical depiction of the interface. The section concludes by introducing the *one-field* formulation of these equations, which is the base for the numerical methods explored in the subsequent sections and chapters of this thesis.

2.1 Single phase fluid dynamics

In the realm of fluid mechanics, the principles of conservation are pivotal in understanding and predicting fluid behavior. The conservation law states that the change in the total amount of a quantity within a specific domain equals the net amount entering or exiting that domain, plus any contributions from sources and/or sinks [20]. The conservation of mass, momentum, and energy together with the material behavior determines the evolution of fluid flows, i.e., fluid dynamics [20].

The conservation of mass in a single-phase fluid flow is governed by

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

where ρ represents mass density. In this thesis, the research focuses on low Mach number flows, for which flow compressibility is negligible. The density of each flow can be considered constant and homogeneous during motion, characterizing it as *incompressible* flow. Under this assumption, the mass conservation equation eq. (2.1) is simplified

$$\nabla \cdot \mathbf{v} = 0, \quad (2.2)$$

which is named the *continuity equation* or the volume conservation equation.

The momentum balance in conservative form reads as

$$\frac{\partial(\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho \mathbf{v} \mathbf{v}) - \nabla \cdot \mathbf{T} = \mathbf{f}_v, \quad (2.3)$$

where \mathbf{v} , \mathbf{T} , and \mathbf{f}_v denote stress tensor, and force density of the body forces, respectively. In the present work, the force per unit volume \mathbf{f}_v is solely attributed to gravitational force

for single phase flow, i.e, $\mathbf{f}_v = \rho\mathbf{g}$. For Newtonian fluids, the stress \mathbf{T} is modeled as a linear function of the rate of strain, more precisely

$$\mathbf{T} = (-P + \frac{2}{3}\mu\nabla \cdot \mathbf{v})\mathbf{I} + \mu(\nabla\mathbf{v} + (\nabla\mathbf{v})^T) \quad (2.4)$$

with the pressure P , the unit tensor \mathbf{I} , the dynamic viscosity μ . The term of divergence of the stress in eq. (2.3) has the formulation as

$$\nabla \cdot \mathbf{T} = \nabla \cdot \left((-P + \frac{2}{3}\mu\nabla \cdot \mathbf{v})\mathbf{I} + \mu(\nabla\mathbf{v} + (\nabla\mathbf{v})^T) \right). \quad (2.5)$$

The dynamic viscosity μ keeps constant in Newtonian fluids, which results in

$$\nabla \cdot (\mu(\nabla\mathbf{v})^T) = \mu\nabla(\nabla \cdot \mathbf{v}) \quad (2.6)$$

Substituting eq. (2.2) into eq. (2.5) and eq. (2.6) yields

$$\nabla \cdot \mathbf{T} = -\nabla P + \nabla \cdot (\mu\nabla\mathbf{v}) \quad (2.7)$$

for the single phase fluids. Inserting eq. (2.7) into eq. (2.3) gives the final form of the incompressible Navier-stokes equation

$$\frac{\partial(\rho\mathbf{v})}{\partial t} + \nabla \cdot (\rho\mathbf{v}\mathbf{v}) - \nabla \cdot (\mu\nabla\mathbf{v}) = -\nabla P + \rho\mathbf{g}. \quad (2.8)$$

Another premise of this thesis is the negligible impact of temperature variations within the considered flows. This assumption leads to *isothermal* conditions, allowing the decoupling of the energy conservation equation from the momentum conservation equation eq. (2.3). In fact, the energy conservation equation is not addressed in this work.

2.2 Two-phase fluid dynamics with interface

In fig. 1, a basic two-phase flow system is illustrated. The entire domain, denoted as Ω , is separated by the interface $\Sigma(t)$ into two distinct subdomains: $\Omega^-(t)$ and $\Omega^+(t)$. Each subdomain is filled with a single-phase fluid, adhering to eq. (2.2) and eq. (2.8) within its respective region. These flows are considered *immiscible* with the assumption that no phase change happens at the interface. The *assumption of a sharp interface* is adopted in this context,

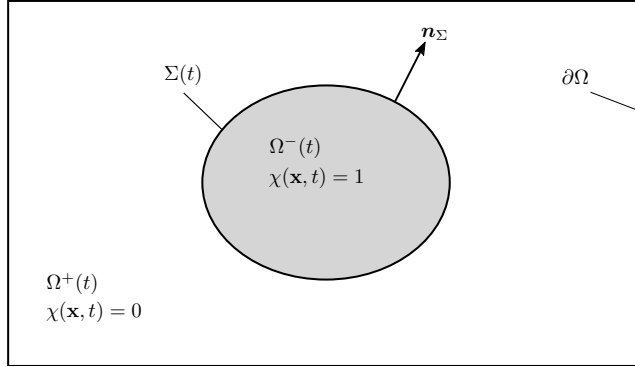


Figure 1: The domain Ω , split by the fluid interface $\Sigma(t)$ into two sub-domains Ω^\pm .

which approximates very thin transition zone between the two bulk phases by a surface of mathematical zero thickness - the sharp interface.

In this thesis, a phase indicator function $\chi(\mathbf{x}, t)$ is adopted to implicitly describe the interface $\Sigma(t)$. As shown in fig. 1, the indicator $\chi(\mathbf{x}, t)$ is defined as

$$\chi(\mathbf{x}, t) := \begin{cases} 1, & \text{for } \mathbf{x} \in \Omega^-(t), \\ 0, & \text{for } \mathbf{x} \in \Omega^+(t), \end{cases} \quad (2.9)$$

which is a discontinuous function. The interface is characterized by a sharp transition in the χ field, where an abrupt change from one value to another marks the interface location. Since the absence of phase change is assumed in this study, each fluid element retains its phase indicator value during its motion.

An approach that employs a single set of governing equations applicable to the entire flow domain, encompassing all the phases, is adopted in this thesis. The so-called one-field approach treats the phases as a single fluid but with material properties changing abruptly at the interface. Considering the indicator function, the flow density and viscosity in the full domain can be written as

$$\rho(\mathbf{x}, t) = \rho^- \chi(\mathbf{x}, t) + \rho^+ (1 - \chi(\mathbf{x}, t)) = (\rho^- - \rho^+) \chi(\mathbf{x}, t) + \rho^+, \quad (2.10)$$

$$\mu(\mathbf{x}, t) = \mu^- \chi(\mathbf{x}, t) + \mu^+ (1 - \chi(\mathbf{x}, t)) = (\mu^- - \mu^+) \chi(\mathbf{x}, t) + \mu^+. \quad (2.11)$$

A singular force exerted by the interface is *surface tension*, which describes an inclination for the interface to minimize its free energy. The surface tension force density is expressed as follows

$$\mathbf{f}_\Sigma = \sigma \kappa \mathbf{n}_\Sigma \delta_\Sigma + \nabla_\Sigma \sigma, \quad (2.12)$$

where σ is the surface tension coefficient, $\kappa = \nabla_\Sigma \cdot (-\mathbf{n}_\Sigma)$ indicates the interface curvature, i.e., the sum of the two principal curvatures, and ∇_Σ denotes the surface gradient. To accord surface tension present solely at the interface with Control Volume (CV)-integrity in Finite Volume (FV) method, the fundamental method in this work and elaborated in section 3.1, the singular surface tension term is formulated via an *interfacial Dirac distribution* (denoted by $\delta_\Sigma(\mathbf{x}, t)$), which converts the interfacial force to a volume force. For an arbitrary function f which has definition at interface Σ that intersects a CV, e.g., Ω_c with $\tilde{\Sigma}$ [21], it gives

$$\int_{\Omega_c} \delta_\Sigma(\mathbf{x}) f(\mathbf{x}) dV = \int_{\tilde{\Sigma}} f(\mathbf{x}) dS. \quad (2.13)$$

Within the framework of this thesis, it is assumed that the surface tension coefficient remains constant. As a result, the second term in eq. (2.12), involving the gradient of the surface tension coefficient, vanishes.

In the two phase fluids, the viscosity varies in a domain intersected by a interface, regarding to eq. (2.11), which causes that $\nabla \cdot (\mu(\nabla \mathbf{v})^\top)$ is unequal to 0 and should be restored in eq. (2.5). By incorporating eqs. (2.10) to (2.12) and $\nabla \cdot (\mu(\nabla \mathbf{v})^\top)$ in the single-phase momentum conservation equation eq. (2.8), it is possible to adapt this equation for a two-phase flow system. The extended formulation is as follows:

$$\frac{\partial(\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho \mathbf{v} \mathbf{v}) - \nabla \cdot (\mu (\nabla \mathbf{v} + (\nabla \mathbf{v})^\top)) = -\nabla P + \rho \mathbf{g} + \sigma \kappa \mathbf{n}_\Sigma \delta_\Sigma. \quad (2.14)$$

This equation integrates the essential dynamics of two-phase flow, encompassing both phases' momentum conservation while accommodating the unique characteristics of the fluid interface.

In this thesis, a modified pressure p is utilized, defined by subtracting the hydrostatic pressure from the total pressure P . This is mathematically expressed as:

$$p = P - \rho \mathbf{g} \cdot \mathbf{x}$$

where \mathbf{x} represents the position vector and \mathbf{g} is constant gravitational acceleration. Accordingly, the pressure and gravitational terms in eq. (2.14) are reformulated as:

$$\begin{aligned}
-\nabla P + \rho \mathbf{g} &= -\nabla P + \nabla(\rho \mathbf{g} \cdot \mathbf{x}) - (\mathbf{g} \cdot \mathbf{x}) \nabla \rho \\
&= -\nabla(P - \rho \mathbf{g} \cdot \mathbf{x}) - (\mathbf{g} \cdot \mathbf{x}) \nabla \rho \\
&= -\nabla p - (\mathbf{g} \cdot \mathbf{x}) \nabla \rho.
\end{aligned}$$

This modification of pressure is advantageous for establishing a force-balanced model across the interface, a topic further elaborated in section 9.6. The *one-field* formulation of the momentum equation is thus

$$\frac{\partial(\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho \mathbf{v} \mathbf{v}) - \nabla \cdot (\mu (\nabla \mathbf{v} + (\nabla \mathbf{v})^\top)) = -\nabla p - (\mathbf{g} \cdot \mathbf{x}) \nabla \rho + \sigma \kappa \mathbf{n}_\Sigma \delta_\Sigma. \quad (2.15)$$

Equation (2.15) and the continuity equation eq. (2.2) constitute the core mathematical model for two-phase flow mechanics within this thesis.

3 Numerical method

This section delves into the FV method employed in this study for solving the mathematical models of incompressible, immiscible two-phase flow systems, outlined in the preceding section 2. Section 3.1 introduces briefly the FV method. Section 3.2 offers an overview of the spatial decomposition of the computational domain and the geometric characterization of the fundamental computational element, named cell. In Section 3.3, attention shifts to the process of discretizing each term in eq. (2.15). The FV method is applied to break down the equation into discrete counterparts, suitable for numerical computation. Section 3.4 turns to the discussion of a family of iterative methods employed to address the challenge of decoupling pressure-velocity linkage in the incompressible Navier-Stokes equations.

The FV method is applied in various styles across different platforms. In this study, the focus is on the specific implementations within the OpenFOAM® platform. Therefore, all subsequent introductions and explanations are tailored to align with OpenFOAM's methodologies. For more comprehensive details, readers are encouraged to consult the relevant literature, such as [17, 19, 22]

3.1 Finite Volume Method

The finite-volume (FV) method directly discretizes the integral form of conservation equations. It achieves this by subdividing the computational domain into finite, contiguous, and non-overlapping control volumes and integrating the equations over control volumes, ensuring the exact conservation of relevant properties within each volume. Second-order accurate volume averages are associated as solution variables with volume centroids. The interpolation extends values to the volume surfaces from the centroids. This interpolation, coupled with suitable quadrature formulae, approximates both surface and volume integrals. Algebraic equations are then derived for each control volume, incorporating neighboring CV averages.

The finite-volume method is compatible with various mesh types, including structured and unstructured meshes. Unstructured meshes, in particular, offer enhanced flexibility in handling complex geometries. As the method operates solely on control volume boundaries, it maintains conservation properties as long as the surface integrals applied at these boundaries

align with the control volumes sharing the boundary [23].

Three levels of approximation are required for the family of FV methods [16]: *integration*, *differentiation* and *interpolation*.

3.1.1 Integration

The cornerstone of the finite volume method is the *control volume integration*. A generic spatially-varying quantity $\phi(\mathbf{x})$ has a volumetric integral approximated by the value $\phi(\mathbf{x}_c)$ at the geometric centroid of CV Ω_c as

$$\int_{\Omega_c} \phi(\mathbf{x}) dV = |\Omega_c| \phi(\mathbf{x}_c) + O(|\mathbf{x} - \mathbf{x}_c|^2), \quad (3.1)$$

with second-order accuracy according to the Taylor Series expansion within Ω_c , given by

$$\phi(\mathbf{x}) = \phi(\mathbf{x}_c) + (\mathbf{x} - \mathbf{x}_c) \cdot (\nabla \phi)_c + O(|\mathbf{x} - \mathbf{x}_c|^2), \quad (3.2)$$

and the definition of the centroid of Ω_c according to

$$\int_{\Omega_c} (\mathbf{x} - \mathbf{x}_c) dV = 0. \quad (3.3)$$

Similarly, the face integrals can be evaluated by

$$\int_f \phi(\mathbf{x}) dS = |\mathbf{S}_f| \phi_f + O(|\mathbf{x} - \mathbf{x}_f|^2) \quad (3.4)$$

with the definition of face centroid \mathbf{x}_f according to

$$\int_f d\mathbf{S} (\mathbf{x} - \mathbf{x}_f) = 0, \quad (3.5)$$

where \mathbf{S}_f indicates the face area normal of the face f with the magnitude $|\mathbf{S}_f|$ denoting the face area and normal $\mathbf{S}_f/|\mathbf{S}_f|$ directed outwards.

3.1.2 Differentiation

In a control volume, the bounding face values are directly linked to the discretization of the first- and second-order differentiation [23]. For both differentiation, the *Gauss's divergence theorem* can be applied to the volume integral. The gradient of $\phi(\mathbf{x})$ is exemplified to discretize the volume integral of the first-order differentiation within Ω_c , which is written as

$$\begin{aligned}\int_{\Omega_c} \nabla \phi dV &= \oint d\mathbf{S} \phi \\ &= \sum_{f \in F_c} \int_f d\mathbf{S} \phi \\ &= \sum_{f \in F_c} \phi_f \mathbf{S}_f + e(h^2),\end{aligned}\tag{3.6}$$

by utilizing the Gauss-divergence theorem, where F_c represents the set of boundary faces of cell Ω_c and h is a characteristic discretization length. It's assumed that $\phi(\mathbf{x})$ is second-order differentiable. The volume integral of the second derivative of $\phi(\mathbf{x})$ can be expressed as follows

$$\begin{aligned}\int_{\Omega_c} \nabla \cdot \nabla \phi dV &= \oint d\mathbf{S} \cdot (\nabla \phi) \\ &= \sum_{f \in F_c} \int_f d\mathbf{S} \cdot (\nabla \phi) \\ &= \sum_{f \in F_c} \mathbf{S}_f \cdot (\nabla \phi)_f + e(h^2).\end{aligned}\tag{3.7}$$

The quantity of $\phi(\mathbf{x})$ can be replaced by a vector, such as the velocity \mathbf{v} . In this case, the discretizations of the volumetric integrals have a very similar formulation as eqs. (3.6) and (3.7), which are discussed in section 3.3.

3.1.3 Interpolation

The approximations of the integrals in eqs. (3.6) and (3.7) require the values of variables at locations other than computational nodes (CV centers), namely ϕ_f and $(\nabla \phi)_f$. The process of estimating values of a physical quantity at locations within a computational domain based

on known values at computational nodes is named *interpolation*. Numerous schemes are available for calculating ϕ and its gradient at the cell face from cell centroid values [16, 24]. An exemplified straightforward approximation for the value at the CV-face center is a linear interpolation (equivalent to Central Differencing Scheme (CDS)) between the values of centers adjacent to \mathbf{S}_f . The scheme is represented as

$$\phi_f = \lambda\phi_{N_f} + (1 - \lambda)\phi_{O_f} \quad (3.8)$$

$$(\nabla\phi)_f = \lambda(\nabla\phi)_{N_f} + (1 - \lambda)(\nabla\phi)_{O_f} \quad (3.9)$$

where N_f, O_f note the neighbor cell and owner cell of the face f , the description of neighbor and owner cell referring to section 3.2.2, and λ is the linear interpolation factor defined by

$$\lambda = \frac{|\mathbf{x}_f - \mathbf{x}_{O_f}|}{|\mathbf{x}_{N_f} - \mathbf{x}_{O_f}|}. \quad (3.10)$$

For the face normal gradient of the implicit variables like pressure in conservation equations, an approximation method using *compact stencil* is adopted as

$$(\nabla\phi)_f \cdot \frac{\mathbf{S}_f}{|\mathbf{S}_f|} = \frac{\phi_{N_f} - \phi_{O_f}}{|\mathbf{x}_{N_f} - \mathbf{x}_{O_f}|} \quad (3.11)$$

to avoid involving large neighboring stencils in the calculation, which improves the computational efficiency when solving large sparse linear equation systems [19]. The compact stencil method is sensitive to mesh quality, specifically the non-orthogonality and skewness of mesh, discussed in section 3.2.3. It necessitates numerical corrections for poor mesh quality, addressed in section 9.6.

3.2 Spatial discretization

The process of spatially discretizing the physical domain involves creating a mesh, which serves as the basis for solving the conservation equations. This entails dividing the entire domain into distinct, non-overlapping subdomains (*cells*) that collectively encompass the entire computational domain, forming a mesh system. Many techniques are employed to generate mesh [25], leading to various mesh types. Mesh generation is beyond the scope of this thesis.

3.2.1 Geometric information

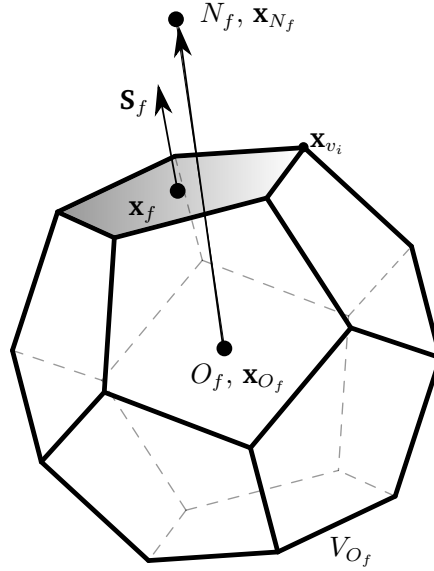


Figure 2: Exemplary general mesh cell Ω_{O_f} .

Figure 2 depicts a general mesh cell Ω_{O_f} , which is bounded by a set of polygons, named *faces*. Each face is a sequence of *indices* of vertices. The geometric information, including the position of vertices, cell centers, face centers, face area normals, and cell volumes, is pivotal for the FV method [16].

Once the mesh is generated, the position vectors of all its vertices are fixed. From these positions, the remaining geometric information can be deduced. It is important to note, however, that in cases of complex geometries, only tetrahedral meshes ensure perfectly planar faces. For other polyhedral shapes, non-planarity may occur, complicating the computation of further geometric information, as noted by [22]. The acquisition of the geometric information in OpenFOAM[®] is exemplified by the cell Ω_{O_f} shown in fig. 2.

An intermediate face center is first obtained regarding

$$\mathbf{x}'_f = \frac{1}{N_f} \sum_{i=1}^{N_f} \mathbf{x}_{v_i}, \quad (3.12)$$

with a digit N_f indicating the number of vertices belonged to the face f . The \mathbf{S}_f in fig. 2 represents the area normal, i.e. a normal with the magnitude of the face area $|\mathbf{S}_f|$ and face normal direction. The calculation of the face area normal relies on summing the area normal of sub-triangles constructed by \mathbf{x}'_f and edges of the face polygon, shown as

$$\mathbf{S}_f = \frac{1}{2} \sum_{i=1}^{N_f-1} (\mathbf{x}_{v_{i+1}} - \mathbf{x}_{v_i}) \times (\mathbf{x}'_f - \mathbf{x}_{v_i}) \quad (3.13)$$

An area-weighted method is used to estimate the final face center, which has the form

$$\mathbf{x}_f = \frac{1}{|\mathbf{S}_f|} \sum_{i=1}^{N_f-1} \left| \left(\frac{1}{2} (\mathbf{x}_{v_{i+1}} - \mathbf{x}_{v_i}) \times (\mathbf{x}'_f - \mathbf{x}_{v_i}) \right) \right| \left(\frac{1}{3} (\mathbf{x}'_f + \mathbf{x}_{v_i} + \mathbf{x}_{v_{i+1}}) \right). \quad (3.14)$$

The cell center \mathbf{x}_{O_f} and volume V_{O_f} are calculated from face center and face area fraction by *Gauss' s theorem*. Two preliminary equations are given:

$$\begin{aligned} \nabla \cdot \mathbf{x} &= 3, \\ \nabla |\mathbf{x}|^2 &= 2\mathbf{x}. \end{aligned}$$

The cell volume $|\Omega_{O_f}|$ can be approximated as

$$\begin{aligned} |\Omega_{O_f}| &= \int_{\Omega_{O_f}} dV = \frac{1}{3} \int_{\Omega_{O_f}} \nabla \cdot \mathbf{x} dV \\ &= \frac{1}{3} \int_f d\mathbf{S} \cdot \mathbf{x} \\ &\approx \frac{1}{3} \sum_f \mathbf{S}_f \cdot \mathbf{x}_f. \end{aligned}$$

The similar derivation for cell center \mathbf{x}_{O_f} reads as

$$\begin{aligned} \mathbf{x}_{O_f} &= \frac{1}{|\Omega_{O_f}|} \int_{\Omega_{O_f}} \mathbf{x} dV = \frac{1}{2|\Omega_{O_f}|} \int_{\Omega_{O_f}} \nabla |\mathbf{x}|^2 dV \\ &= \frac{1}{2|\Omega_{O_f}|} \int_f d\mathbf{S} |\mathbf{x}|^2 \\ &\approx \frac{1}{2|\Omega_{O_f}|} \sum_f \mathbf{S}_f |\mathbf{x}_f|^2. \end{aligned}$$

3.2.2 Mesh structure

The FV method is favored partly due to its adaptability to various mesh types, including both structured and unstructured meshes. Structured meshes consist of intersecting line families, with each mesh point located at the intersection of a single line from each family. This pattern applies to both 2D (two line families) and 3D (three line families) structures. In contrast, unstructured meshes comprise arbitrarily distributed points connected by triangles, quadrilaterals, or polygons in 2D, and by a range of polyhedrals like tetrahedra, prisms, pyramids, hexahedra, or other polyhedrals in 3D [20]. This diversity in mesh structure affects the discretization approach for both the domain and equations, as well as the connectivity and addressing of mesh cells [19]. Unstructured meshes are particularly advantageous for their flexibility in fitting complex shapes and boundaries, providing a more precise representation of the physical domain. Additionally, they support adaptive mesh refinement, which is critical for enhancing accuracy in areas with sharp gradients or where greater detail is necessary. Due to these benefits, OpenFOAM® utilizes unstructured meshes [19].

The mesh connectivity and addressing in OpenFOAM® are introduced briefly henceforth. The mesh connectivity determines how and which (*stencil*) cells are accessed and, furthermore, calculated. The FV method relies on mesh connectivity to compute fluxes through faces, gradient calculations, reconstructions, and other operations central to solving the governing equations. Three important types of mesh connectivity [19] is listed below.

- *Vertex-to-Cell Connectivity*: This defines which cells are connected to a given vertex. It's important for reconstructing cell data from vertex data and vice versa.
- *Face-to-Cell Connectivity*: Critical for the flux calculations across cell boundaries, this connectivity defines which cells are adjacent to each other, via shared faces.
- *Boundary Connectivity*: Defines how cells and faces interact with the boundaries of the computational domain, which could be walls, inlets, outlets, etc.

There are other connectivity types, such as *Vertex-to-Edge Connectivity* and *Edge-to-Face Connectivity*, which are used rarely and not described here.

Mesh addressing involves the use of data structures to reference and access various mesh entities (like cells, faces, and vertices). These data structures are crucial for efficient

computation, as they enable the software to locate and manipulate the entities of the mesh quickly. Addressing is closely linked to mesh connectivity, as it provides the means to traverse and reference the connectivity information. OpenFOAM[®] employs various addressing methods, such as *indirect addressing*, *owner-neighbor addressing*, and *boundary mesh addressing* [19].

An illustrative example of mesh addressing in unstructured meshes within OpenFOAM[®] is depicted in fig. 2. Here, a mesh face f of a cell is associated with an index pair (O_f, N_f) , comprising the indices O_f, N_f of the adjacent cells Ω_{O_f} and Ω_{N_f} . The face-adjacent cell with the lower index is designated as the 'owner' cell (O_f), while the cell with the higher index is termed the 'neighbor' cell (N_f). Notably, the area normal \mathbf{S}_f consistently points from the owner cell towards the neighbor cell.

3.2.3 Mesh quality

After the generation of the mesh, the mesh quality should be assessed. It influences the accuracy, stability, and efficiency of all numerical methods. Good mesh quality is crucial for obtaining reliable and accurate simulation results. Hereafter, four common mesh quality metrics are described below with corresponding illustrations in fig. 3.

- *Orthogonality*: Refers to the angle θ_f between the face area normal \mathbf{S}_f and the direction of \mathbf{d} , a vector connecting the centers of two adjacent cells sharing the face, i.e, $\mathbf{d} = \mathbf{x}_{N_f} - \mathbf{x}_{O_f}$, in fig. 3a. For non-orthogonal mesh, the angle θ_f is not equal to zero. The non-orthogonality reduces the accuracy of the central difference approximation to the derivative in the direction of the face normal [16]. In this study, an iterative correction method to alleviate the errors from non-orthogonality is introduced in section 9.6.
- *Skewness*: Refers to the deviation of $\mathbf{x}_{f'}$, the intersection point between \mathbf{d} and face, from the face center \mathbf{x}_f , in fig. 3b. The common practice to estimate the value at the cell face is to use a linear interpolation profile. To keep the overall accuracy of the spatial discretization method second order, all face integrations need to take place at point \mathbf{x}_f [17]. For a skew mesh, the linearly interpolated face value at $\mathbf{x}_{f'}$ does not coincide with the cell center value at \mathbf{x}_f , which the loss of the second-order accuracy.

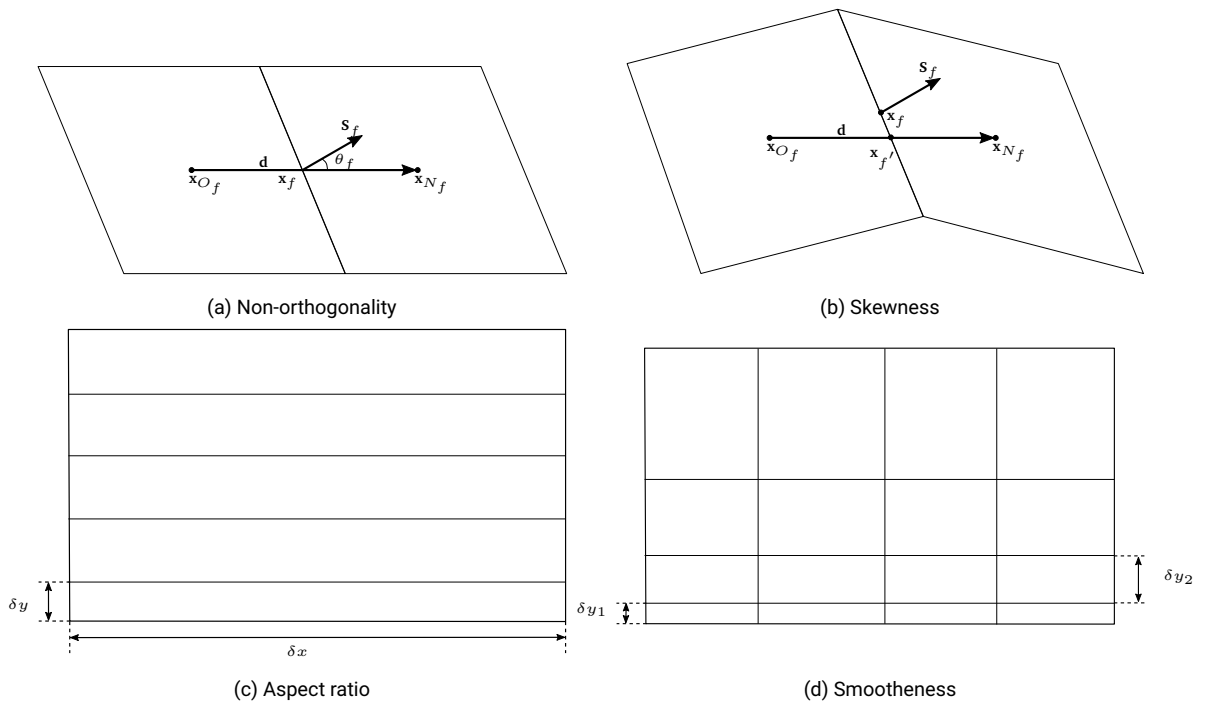


Figure 3: Four mesh quality metrics

- *Aspect Ratio*: The ratio of the longest to the shortest length of a cell, namely $\frac{\delta x}{\delta y}$ in fig. 3c. A high aspect ratio can smear gradients and cause inaccurate interpolation across cells.
- *Smoothness*: Refers to the transition in size between contiguous cells, i.e., $\frac{\delta y_2}{\delta y_1}$ in fig. 3d, which is also known as expansion rate, growth factor, or uniformity. A large rate of cell expansion adds numerical diffusion to the solution.

The errors from non-orthogonality and skewness can be corrected with special numerical techniques. However, when the mesh is not uniform enough, i.e. with the large aspect ratio and or the expansion rate, remeshing is usually necessary [16].

3.2.4 Variable arrangement

Meshes used in computational fluid dynamics are classified based on the locations where variables from conservation equations are stored. There are two primary categories: *collocated mesh* and *staggered mesh*. In a collocated mesh, all variables, such as velocity, pressure, and density, are stored at a single set of points within each cell, typically at the cell center. On the other hand, the staggered mesh employs a different storage approach. Here, vector variables like velocity are calculated and stored at the cell faces, whereas scalar variables such as pressure and density are computed and stored at the cell centers [17]. This configuration is particularly effective in suppressing unphysical oscillations in the pressure and velocity fields, especially under strong coupling conditions between these fields. It also ensures the conservation of kinetic energy approximation [16]. However, managing a staggered mesh, especially for unstructured meshes, can be complex due to the intricate bookkeeping required for each variable's location [18].

Despite the benefits of staggered meshes in certain scenarios, the collocated arrangement is generally preferred in modern codes due to its simpler data structure and easier implementation. Moreover, recent advancements in numerical algorithms have significantly reduced problems that were once common in collocated grids, such as pressure-velocity decoupling [16]. OpenFOAM[®], the platform used in this study, employs the collocated mesh arrangement, benefiting from these advancements and the simplicity of the collocated data structure.

3.3 Equation discretization

In Section 2, the mathematical model for fluid mechanics governing two-phase flow in one-fluid formulation is derived. In this subsection, the focus is on discretizing this model using FV method on collocated unstructured mesh. The continuity equation eq. (2.2) and momentum

conservation eq. (2.15) with named terms equation are re-presented here in integral form

$$\begin{aligned}
& \int_{\Omega_c} \nabla \cdot \mathbf{v} dV = 0, \\
& \underbrace{\int_{\Omega_c} \frac{\partial(\rho \mathbf{v})}{\partial t} dV}_{\text{Transient term}} + \underbrace{\int_{\Omega_c} \nabla \cdot (\rho \mathbf{v} \mathbf{v}) dV}_{\text{Convective term}} - \underbrace{\int_{\Omega_c} \nabla \cdot (\mu (\nabla \mathbf{v} + (\nabla \mathbf{v})^T)) dV}_{\text{Viscous term}} \\
& \qquad \qquad \qquad = \underbrace{\int_{\Omega_c} (-\nabla p - (\mathbf{g} \cdot \mathbf{x}) \nabla \rho + \sigma \kappa \mathbf{n}_\Sigma \delta_\Sigma) dV}_{\text{Source term}}.
\end{aligned} \tag{3.15}$$

The discretization of both equations is introduced in the following subsections.

3.3.1 Continuity equation

Applying the divergence theorem introduced in section 3.1.2 to the continuity equation results in

$$\begin{aligned}
\int_{\Omega_c} \nabla \cdot \mathbf{v} dV &= \oint \mathbf{v} \cdot d\mathbf{S} \\
&= \sum_{f \in F_c} \int_f \mathbf{v} \cdot d\mathbf{S} \\
&= 0, \\
\sum_{f \in F_c} \int_f \mathbf{v} \cdot d\mathbf{S} &\approx \sum_{f \in F_c} \mathbf{v}_f \cdot \mathbf{S}_f = \sum_{f \in F_c} F_f,
\end{aligned} \tag{3.16}$$

where $d\mathbf{S}$ denotes the infinite area normal of face f , this face area normal is defined by $\mathbf{S}_f = \int_f d\mathbf{S}$. The *volumetric flux* across f is defined by

$$F_f := \mathbf{v}_f \cdot \mathbf{S}_f. \tag{3.17}$$

The flux across a face f is consistent in magnitude and differs only in sign between two cells sharing it. This is essential for maintaining mass conservation properties at the discrete level. It's notable that the discretized continuity equation is not approximated and solved, but used as a constraint to decouple the pressure-velocity coupling and as a metric to assess the convergence of the computational solution [26].

3.3.2 Convective term

The convective term is discretized using the divergence theorem introduced in section 3.1.2

$$\begin{aligned}
 \int_{\Omega_c} \nabla \cdot (\rho \mathbf{v} \mathbf{v}) dV &= \sum_{f \in F_c} \int_f \rho \mathbf{v} \mathbf{v} \cdot d\mathbf{S} \\
 &= \sum_{f \in F_c} \int_f \mathbf{v} (\rho_f \mathbf{v} \cdot d\mathbf{S}) \\
 &\approx \sum_{f \in F_c} (\rho_f \mathbf{v}_f \cdot \mathbf{S}_f) \mathbf{v}_f = \sum_{f \in F_c} (\rho_f F_f) \mathbf{v}_f
 \end{aligned} \tag{3.18}$$

with a second-order midpoint approximation of velocity, i.e., \mathbf{v}_f , at the face center \mathbf{x}_f , regarding section 3.1.1. It is noticeable that the convection term $(\rho \mathbf{v} \mathbf{v})$ is non-linear. This issue is solved by linearization which applies the newest updated known velocity to $(\rho \mathbf{v}_f \cdot \mathbf{S}_f)$. The approximated *mass flux* across f are defined here as

$$\dot{m} = \rho_f \mathbf{v}_f \cdot \mathbf{S}_f = \rho_f F_f. \tag{3.19}$$

Within the collocated FV method, the face centers are not the computational nodes, making face-centered quantities like \mathbf{v}_f unavailable. Therefore, an interpolation to \mathbf{v}_f is entailed.

3.3.3 Viscous term

In the case of the diffusion term, the discretization is carried out with the divergence theorem as

$$\begin{aligned}
 \int_{\Omega_c} \nabla \cdot (\mu (\nabla \mathbf{v} + (\nabla \mathbf{v})^T)) dV &= \sum_{f \in F_c} \int_f \mu (\nabla \mathbf{v} + (\nabla \mathbf{v})^T) \cdot d\mathbf{S} \\
 &\approx \sum_{f \in F_c} \mu_f ((\nabla \mathbf{v})_f \cdot \mathbf{S}_f + (\nabla \mathbf{v})_f^T \cdot \mathbf{S}_f)
 \end{aligned} \tag{3.20}$$

The gradient flux $(\nabla \mathbf{v})_f \cdot \mathbf{S}_f$ is treated implicitly and approximated as stated in eq. (3.11), more specifically

$$(\nabla \mathbf{v})_f \cdot \mathbf{S}_f \approx \frac{\mathbf{v}_{N_f} - \mathbf{v}_{O_f}}{|\mathbf{x}_{N_f} - \mathbf{x}_{O_f}|} |\mathbf{S}_f|. \tag{3.21}$$

More numerical techniques to discretize the gradient flux on a non-orthogonal mesh are extensively reviewed in section 9.6.

On the other hand, the transposed tensor $(\nabla \mathbf{v})_f^T$ is addressed explicitly and approximated following eq. (3.9), e.g. with CDS, having the form

$$(\nabla \mathbf{v})_f^T \approx \lambda (\nabla \mathbf{v})_{N_f}^T + (1 - \lambda) (\nabla \mathbf{v})_{O_f}^T. \quad (3.22)$$

The divergence theorem is applied to explicitly approximate the gradient at the cell center. For the owner cell Ω_{O_f} , $(\nabla \mathbf{v})_{O_f}$ is computed by

$$\begin{aligned} (\nabla \mathbf{v})_{O_f} &= \frac{1}{|\Omega_{O_f}|} \int_{\Omega_{O_f}} \nabla \mathbf{v} dV \\ &\approx \frac{1}{|\Omega_{O_f}|} \sum_{f \in F_{O_f}} \mathbf{v}_f \otimes \mathbf{S}_f, \end{aligned} \quad (3.23)$$

where \mathbf{v}_f is interpolated from the explicit center values of saddling cells. An additional approach to calculate the gradient term is Least Squares Fit (LSF), which offers more flexibility concerning the accuracy and the stencil but requires higher cost for computation [17] as addressed in Appendix 2. The $(\nabla \mathbf{v})_f^T$ is then obtained by transposing $(\nabla \mathbf{v})_{O_f}$.

3.3.4 Source term

Both implicit and explicit discretization for the first derivative at cell centers and faces are discussed in Sections 3.1.2 and 3.3.3. The pressure and density gradient terms in the source term in eq. (3.15) are discretized following these discretization schemes. The sole remaining term is the surface tension force $\sigma \kappa \mathbf{n}_\Sigma \delta_\Sigma$. In this study, the Continuum Surface Force (CSF) model [27] is deployed to estimate the surface tension as follows:

$$\mathbf{f}_\Sigma = \sigma \kappa \mathbf{n}_\Sigma \delta_\Sigma \approx \sigma \kappa \nabla c(\mathbf{x}), \quad (3.24)$$

where $c(\mathbf{x})$ is a characteristic function. The definition of this characteristic function depends on the method used to capture the interface, e.g. volume fraction in VOF method or signed distance function in LSM. These interface capturing methods are addressed in section 3.5.

To eliminate the *spurious currents* appearing artificially in what should be a static two-phase flow system, the discretization scheme for the three gradients, i.e. ∇p , $\nabla \rho$, and ∇c

must be consistent, which is called *well-balance* approach. More details about the concept of a well-balanced discretization are discussed in section 9.6.

3.3.5 Transient term

Time is an additional solution dimension for transient flow. Similar to the subdivision of computational domain to CV, the prescribed time interval $[t^0, t^E]$, where t^0, t^E are the start and end of the simulation, is decomposed as a partition $[t^0, t^1, t^2, \dots, t^{n-1}, t^n, t^{n+1}, \dots, t^E]$, such that $t^{n-1} < t^n < t^{n+1}$ with a time interval $\Delta t^{n+1} = t^{n+1} - t^n$ called a time step.

The volume integral of the transient term aligns with the general integral form in Section 3.1.1, stated as follows

$$\int_{\Omega_c} \frac{\partial(\rho \mathbf{v})}{\partial t} dV \approx |\Omega_c| \frac{\partial(\rho_c \mathbf{v}_c)}{\partial t} \quad (3.25)$$

with second-order spatial accuracy. The temporal integral in a time step $[t^n, t^{n+1}]$ of *semi-discrete* eq. (3.15) can be represented as

$$\begin{aligned} \int_{t^n}^{t^{n+1}} \left[|\Omega_c| \frac{\partial(\rho_c \mathbf{v}_c)}{\partial t} + \sum_{f \in F_c} (\rho_f \mathbf{v}_f \cdot \mathbf{S}_f) \mathbf{v}_f - \sum_{f \in F_c} \mu_f ((\nabla \mathbf{v})_f \cdot \mathbf{S}_f + (\nabla \mathbf{v})_f^T \cdot \mathbf{S}_f) \right] dt \\ = \int_{t^n}^{t^{n+1}} \left[|\Omega_c| (-\nabla p - (\mathbf{g} \cdot \mathbf{x}) \nabla \rho + \sigma \kappa \nabla c) \right] dt \quad (3.26) \end{aligned}$$

To shorten the eq. (3.26), a function $f(\mathbf{v}(\mathbf{x}, t))$ including all spatial terms from eq. (3.26) is utilized. The shortened version of eq. (3.26) is given as follows

$$\int_{t^n}^{t^{n+1}} |\Omega_c| \frac{\partial(\rho_c \mathbf{v}_c)}{\partial t} dt = \int_{t^n}^{t^{n+1}} f(\mathbf{v}(\mathbf{x}, t)) dt \quad (3.27)$$

Conducting the temporal integration of eq. (3.27) results in

$$\int_{t^n}^{t^{n+1}} |\Omega_c| \frac{\partial(\rho_c \mathbf{v}_c)}{\partial t} dt = |\Omega_c| \rho_c (\mathbf{v}_c^{n+1} - \mathbf{v}_c^n) = \int_{t^n}^{t^{n+1}} f(\mathbf{v}(\mathbf{x}, t)) dt, \quad (3.28)$$

where the superscripts $n, n+1$ mean the values at time t^n and t^{n+1} , respectively.

Three common temporal approximation methods for the time integration of spatial terms are listed below [16]

- *Explicit Euler method.* In this method, all independent variables \mathbf{v} are treated explicitly, which means that the calculated \mathbf{v} at the end of the last time interval $[t^{n-1}, t^n]$, namely \mathbf{v}^n , is used to update the new \mathbf{v} . The time integration of $f(\mathbf{v}(\mathbf{x}, t))$ is approximated by

$$\int_{t^n}^{t^{n+1}} f(\mathbf{v}(\mathbf{x}, t)) dt \approx f(\mathbf{v}^n) \Delta t. \quad (3.29)$$

This method has the first-order accuracy in time. A few stability tests conducted in [16, 17] show that this method is *conditionally stable*. It is stable only under the condition that the *Courant number* $Co = \frac{|\mathbf{v}| \Delta t}{h}$ is smaller than unity. Here, $|\mathbf{v}|$ is the characteristic velocity, and h is the cell size. The imposed time restriction mandates a greater number of steps to advance the solution in time, potentially resulting in extended simulation times, particularly with a fine mesh.

- *Implicit Euler method.* This method approximates eq. (3.28) using the implicit value of \mathbf{v} in spatial terms, i.e. \mathbf{v}^{n+1} , which is unknown and obtained by solving a system of algebraic equations for each time step. The approximated form is expressed as

$$\int_{t^n}^{t^{n+1}} f(\mathbf{v}(\mathbf{x}, t)) dt \approx f(\mathbf{v}^{n+1}) \Delta t. \quad (3.30)$$

The accuracy of this method is also first-order. In contrast to the explicit Euler method, this scheme requires solving a large coupled set of equations at each time step. However, it offers the advantage of being able to utilize larger time steps, making it advantageous for many scenarios.

- *Crank-Nicolson method.* This method averages the approximations in eqs. (3.29) and (3.30) and has the form

$$\int_{t^n}^{t^{n+1}} f(\mathbf{v}(\mathbf{x}, t)) dt \approx \frac{1}{2} (f(\mathbf{v}^n) + f(\mathbf{v}^{n+1})) \Delta t. \quad (3.31)$$

This scheme features second-order time discretization and achieves stability over a wider Co range, specifically $Co \leq 2$ [16, 17].

In this study, the implicit Euler method is mainly used, and the final discretized momentum equation is represented as

$$\begin{aligned}
|\Omega_c| \frac{(\rho_c \mathbf{v}_c^{n+1} - \rho_c \mathbf{v}_c^n)}{\Delta t} + \sum_{f \in F_c} (\rho_f \mathbf{v}_f^n \cdot \mathbf{S}_f) \mathbf{v}_f^{n+1} - \sum_{f \in F_c} \mu_f ((\nabla \mathbf{v}^{n+1})_f \cdot \mathbf{S}_f + (\nabla \mathbf{v}^n)_f^T \cdot \mathbf{S}_f) \\
= |\Omega_c| (-\nabla p - (\mathbf{g} \cdot \mathbf{x}) \nabla \rho + \sigma \kappa \nabla c)_c^{n+1} \quad (3.32)
\end{aligned}$$

In the two-phase flow system, the interface evolves with the velocity \mathbf{v} . The spatial and temporal discretization and calculation are discussed in section 3.5.4 and section 6.4.

3.4 Solution algorithms for pressure-velocity coupling

The discretized momentum equation (eq. (3.32)) reveals a linear interdependence among velocity, pressure, and the fluid interface (surface tension, gravity). While the velocity field derived from the momentum equation must satisfy the continuity equation (eq. (3.16)), there is no dedicated transport or other equation for the pressure in the context of incompressible, isothermal flow as investigated in this study. This challenge is commonly referred to as *pressure-velocity coupling*. To address this issue, a strategy is employed wherein the pressure field is constructed to ensure that the velocity satisfies the continuity equation. This involves modifying the continuity equation into an equation for pressure, as outlined by [16, 17].

3.4.1 Construction of pressure equation

By applying interpolation and first derivative schemes, Equation (3.32) is reformulated as

$$a_c \mathbf{v}_c^{n+1} = \mathbf{H}(\mathbf{v}_n^{n+1}) - |\Omega_c| (\nabla p)_c^{n+1} + \mathbf{b}_c^{n+1}, \quad (3.33)$$

where

- a_c is a sum of all coefficients of \mathbf{v}_c^{n+1} ,
- $\mathbf{H}(\mathbf{v}_n^{n+1})$ consists of effects from new velocity of neighbor cells and old velocity, expressed as

$$\mathbf{H}(\mathbf{v}_n^{n+1}) = - \sum_{n \in N(\Omega_c)} a_n \mathbf{v}_n^{n+1} + \frac{|\Omega_c|}{\Delta t} \mathbf{v}_c^n + \sum_{f \in F_c} \mu_f ((\nabla \mathbf{v}^n)_f^T \cdot \mathbf{S}_f) \quad (3.34)$$

with the set $N(\Omega_c)$ including all indices of neighbor cells of Ω_c ,

- \mathbf{b}_c^{n+1} contains the body force terms, i.e., $\mathbf{b}_c^{n+1} = |\Omega_c| (-\mathbf{g} \cdot \mathbf{x}) \nabla \rho + \sigma \kappa \nabla c)^{n+1}$.

The new velocity \mathbf{v}_c^{n+1} can be obtained by dividing eq. (3.33) with a_c , have form

$$\mathbf{v}_c^{n+1} = \frac{\mathbf{H}(\mathbf{v}_n^{n+1})}{a_c} - |\Omega_c| \frac{(\nabla p)_c^{n+1}}{a_c} + \frac{\mathbf{b}_c^{n+1}}{a_c}, \quad (3.35)$$

which must obey continuity equation eq. (3.16) for every time step, resulting in

$$\begin{aligned} & \sum_{f \in F_f} \mathbf{v}_f^{n+1} \cdot \mathbf{S}_f \\ &= \sum_{f \in F_f} \left(\mathbf{S}_f \cdot \left(\frac{\mathbf{H}(\mathbf{v}_n^{n+1})}{a_c} \right)_f - |\Omega_c| \left(\frac{1}{a_c} \right)_f \mathbf{S}_f \cdot (\nabla p)_f^{n+1} + \left(\frac{1}{a_c} \right)_f \mathbf{S}_f \cdot \mathbf{b}_f^{n+1} \right) \\ &= 0. \end{aligned} \quad (3.36)$$

A critical issue to be solved is that the momentum and continuity equation reformulated in eq. (3.35) and eq. (3.36) show strong non-linearity because of their dependency on $\mathbf{H}(\mathbf{v}_n^{n+1})$, which is a function of \mathbf{v}_n^{n+1} , unknown velocities of neighbor cells. This non-linearity can be resolved using an iterative procedure. The most widely used approach for iteratively solving coupled equations is the *segregated approach*, where equations are linearised and solved sequentially one after the other [16, 17, 28, 29]. A brief introduction to three extensively utilized algorithms will be presented below.

3.4.2 SIMPLE

Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) was originally put forward by Patankar and Spalding [30] for the calculations of steady flows and then extended to transient calculations. Ever since the pioneering work by them, it has found widespread application in the majority of commercial CFD codes. In this scheme, a guessed pressure field is used to solve the momentum equations. A pressure-correction equation, deduced from the continuity equation, is then solved to obtain a pressure-correction field, which in turn is used to update the velocity and pressure fields. These guessed fields are progressively improved through the iteration process until convergence is achieved for the velocity and pressure fields.

To elaborate on this process, the new velocity and pressure field is first split into the intermediate part (marked by superscript $*$) and the corrector part (marked by superscript $'$), as follows

$$\begin{aligned}\mathbf{v}^{n+1} &= \mathbf{v}^* + \mathbf{v}', \\ p^{n+1} &= p^* + p'.\end{aligned}\tag{3.37}$$

SIMPLE consists of the following steps:

1. *Momentum prediction:* The intermediate velocity is calculated from

$$\mathbf{v}_c^* = \frac{\mathbf{H}(\mathbf{v}_n^*)}{a_c} - |\Omega_c| \frac{(\nabla p^*)_c}{a_c} + \frac{\mathbf{b}_c^{n+1}}{a_c}\tag{3.38}$$

where the intermediate pressure p^* and known velocity contained in a_c and $\mathbf{H}(\mathbf{v}_n^*)$ are guessed at the first iteration, and usually initialized by their old values at the last time step. Notably, \mathbf{v}_c^* does not necessarily satisfy the continuity equation in this step.

2. *Pressure correction:* Substituting eq. (3.37) into eq. (3.35) and then subtracting eq. (3.38) results in

$$\mathbf{v}'_c = \underbrace{\frac{\mathbf{H}(\mathbf{v}'_n)}{a_c}}_{\text{Neglected}} - |\Omega_c| \frac{(\nabla p')_c}{a_c}\tag{3.39}$$

By substituting eq. (3.39) and eq. (3.37) into continuity equation, the Laplacian equation for p' is constructed, as follows

$$-|\Omega_c| \nabla \cdot \left(\frac{(\nabla p')_c}{a_c} \right) = \nabla \cdot \mathbf{v}_c^* - \underbrace{\nabla \cdot \left(\frac{\mathbf{H}(\mathbf{v}'_n)}{a_c} \right)}_{\text{Neglected}}\tag{3.40}$$

The handling of the operator $\mathbf{H}(\mathbf{v}'_n)$ needs discussion at this moment. Given that the velocity corrector \mathbf{v}' remains unknown at this stage, an approximation for $\mathbf{H}(\mathbf{v}'_n)$ is necessary. Upon convergence of the solution for \mathbf{v} and p , the correctors \mathbf{v}' and p' are rendered as zero. Consequently, in the original SIMPLE method, $\mathbf{H}(\mathbf{v}'_n)$ is disregarded for approximation purposes. Consequently, the correctors p' in this iteration can be obtained by solving eq. (3.40). Omitting this term does not compromise the final solution, as its value converges to zero. However, its exclusion influences the convergence trajectory, as its value can be significant at first iterations. This notable magnitude may lead to either

divergence or deceleration in the convergence rate due to an overestimated pressure corrector. To address this challenge, the *under-relaxation* factor λ^p is introduced for pressure updating, expressed as

$$p = p^* + \lambda^p p', \quad 0 < \lambda^p < 1. \quad (3.41)$$

3. *Velocity correction.* By the omission of $\mathbf{H}(\mathbf{v}'_n)$ and the substitution of newly known p' into eq. (3.39), the velocity corrector \mathbf{v}' is calculated. Similar to λ^p , there is an under-relaxation factor λ^v for limiting the velocity corrector. The velocity is updated following

$$\mathbf{v}_c = \mathbf{v}_c^* + \lambda^v \mathbf{v}'_c, \quad 0 < \lambda^v < 1. \quad (3.42)$$

Moukalled, Mangani, Darwish, et al. [17] proved that when $\lambda^v \approx 1 - \lambda^p$, SIMPLE can reach the optimum acceleration rate of convergence.

4. *Convergence estimation and iteration.* The source term on the right hand of eq. (3.40) is commonly regarded as the criteria to terminate the iteration. If the source term is larger than the prescribed residual tolerance, the newly updated pressure p and \mathbf{v} are applied to eq. (3.38) for the next iteration. And the whole procedure is repeated from step 1.

Algorithm 1 Algorithm for a single SIMPLE time step $[t^n, t^{n+1}]$

while not converged **do**

Solve momentum prediction equation implicitly for \mathbf{v}^* ▷ Equation (3.38)

Solve pressure correction equation for p' ▷ Equation (3.40)

Update of \mathbf{v} and p with under-relaxation factors ▷ Equations (3.41) and (3.42)

end while

Update new fields $\mathbf{v}^{n+1} = \mathbf{v}$, $p^{n+1} = p$

3.4.3 PISO

Despite the use of under-relaxation, the rate of convergence of the SIMPLE algorithm remains problem-dependent and researchers sought alternatives for further improvements. One widely used variant is Pressure-Implicit Split-Operator (PISO)[31], which extends SIMPLE with an additional correction step that involves an additional pressure-correction equation to enhance the convergence.

Different from SIMPLE, the momentum prediction step in section 3.4.2 is not contained in the iteration during a time step. Instead, the updated velocity is employed to solve eq. (3.40) and as the intermediate velocity for the next iteration, while the updated pressure is redefined as the new intermediate pressure for the next iteration. PISO algorithm contains the following steps during a time step:

1. Momentum prediction: The first intermediate velocity $\mathbf{v}_c^{*(1)}$ is calculated according to eq. (3.38). The superscripts (i) marks the i th iteration. Same as in SIMPLE, the eq. (3.38) is solved with the guessed pressure and velocity, usually p^n and \mathbf{v}^n .
2. Pressure correction: The pressure corrector is acquired by solving its Laplacian equation regarding eq. (3.40)

$$-|\Omega_c|\nabla \cdot \left(\frac{(\nabla p'^{(i)})_c}{a_c} \right) = \nabla \cdot \mathbf{v}_c^{*(i)}. \quad (3.43)$$

The pressure is updated by

$$p^{(i)} = p^{*(i)} + \lambda^p p'^{(i)}, \quad 0 < \lambda^p < 1. \quad (3.44)$$

3. Velocity correction: The velocity corrector is calculated by solving

$$\mathbf{v}'_c{}^{(i)} = -|\Omega_c| \frac{(\nabla p'^{(i)})_c}{a_c}, \quad (3.45)$$

followed by updating velocity as

$$\mathbf{v}_c^{(i)} = \mathbf{v}_c^{*(i)} + \lambda^v \mathbf{v}'_c{}^{(i)}, \quad 0 < \lambda^v < 1. \quad (3.46)$$

4. Iteration: When i is not equal to the preset iteration number N_{PISO} , the $i + 1$ th iteration begins with

$$\begin{aligned}\mathbf{v}_c^{*(i+1)} &= \mathbf{v}_c^{(i)}, \\ p^{*(i+1)} &= p^{(i)}.\end{aligned}\tag{3.47}$$

Algorithm 2 Algorithm for a single PISO time step $[t^n, t^{n+1}]$

```
Solve momentum prediction equation implicitly for  $\mathbf{v}^{*(1)}$  ▷ Equation (3.38)
for  $i = 1; i \leq N_{PISO}; ++ i$  do
  Solve pressure correction equation for  $p^{(i)}$  ▷ Equation (3.43)
  Update of  $\mathbf{v}^{(i)}$  and  $p^{(i)}$  with under-relaxation factors ▷ Equations (3.44) and (3.46)
  if  $i \neq N_{PISO}$  then
    Update intermediate pressure and velocity ▷ Equation (3.47)
  else
    if  $i = N_{PISO}$  then
      Update new fields  $\mathbf{v}^{n+1} = \mathbf{v}^{(i)}, p^{n+1} = p^{(i)}$ 
    end if
  end if
end for
```

In the case $N_{PISO} = 1$, PISO is equivalent to one iteration of SIMPLE, which means that the same approximation error regarding neglecting $\mathbf{H}(\mathbf{v}'_n)$ is generated. In the original PISO[31], the authors indicate that at least $N_{PISO} = 3$ are required to converge the solution.

3.4.4 PIMPLE

Piso + siMPLE (PIMPLE) is a solution algorithm available in OpenFOAM[®]. This algorithm combines internal iterations for pressure correction in PISO with the whole iterations for momentum prediction and pressure correction in SIMPLE and adds the third layer iterations for non-orthogonality correction. The non-orthogonality correction is the main topic of section 9.6, which is addressed later. Different from in SIMPLE, the advancement of outer

iteration is controlled by both the prescribed tolerance and the fixed iteration number in PIMPLE.

Algorithm 3 Algorithm for a single PIMPLE time step $[t^n, t^{n+1}]$

```

1:  $o := 1$ 
2: while  $o \leq N_{outer}$  or  $res > tol$  do
3:   Solve momentum prediction equation implicitly for  $\mathbf{v}^{*(o)}$  ▷ Equation (3.38)
4:   for  $i = 1; i \leq N_{inner}; ++i$  do
5:     Solve pressure correction equation for  $p^{(i)}$  ▷ Equation (3.43)
6:     Update of  $\mathbf{v}^{(i)}$  and  $p^{(i)}$  with under-relaxation factors ▷ Equations (3.44) and (3.46)
7:     if  $i \neq N_{inner}$  then ▷ Equation (3.47)
8:       Update intermediate pressure and velocity
9:     else
10:      if  $i = N_{inner}$  then
11:        Update fields  $\mathbf{v}^{o+1} = \mathbf{v}^{(i)}$ ,  $p^{o+1} = p^{(i)}$  for the next outer loop
12:      end if
13:    end if
14:  end for
15:   $++o$ 
16: end while
17: Update fields  $\mathbf{v}^{n+1} = \mathbf{v}^{(o)}$ ,  $p^{n+1} = p^{(o)}$  for the next time step
18:

```

The PIMPLE is one of the most widely used algorithms for transient problems in OpenFOAM®. It is also the basis of the solution algorithm in this study. The combination of PISO and PIMPLE makes it possible to use larger Courant numbers ($Co \gg 1$)[32] in the computation. Comparing algorithm 3 with algorithm 2 and algorithm 1, it is evident that PIMPLE is equivalent to PISO with $N_{outer} = 1$, and equivalent to SIMPLE with $N_{inner} = 1$.

3.4.5 SAAMPLE

A novel segregated solution algorithm, termed SAAMPLE, has been developed by Tolle, Bothe, and Marić [2] to stabilize two-phase flows dominated by surface tension force. Differing

from the classical PIMPLE, SAAMPLE is an iterative approach driven by solution accuracy. In addition to the prescribed inner and outer iteration numbers, two additional convergence conditions are introduced for the outer and inner loops, respectively. For the outer loop, a boolean flag, named $\text{conv}(F_f)$, is defined as follows:

$$\text{conv}(F_f)^o = \begin{cases} true, & \text{if } L_\infty(|F_f - F_f^{prev}|/L_\infty(F_f)) < \text{tol}_{rel} \\ true, & L_\infty(|F_f - F_f^{prev}|) < \text{tol}_{abs} \\ false, & \text{otherwise} \end{cases} \quad (3.48)$$

where the parameters tol_{rel} and tol_{abs} are prescribed for the relative and absolute change of volumetric flux F_f between two consecutive outer iterations. The inner loop performs the pressure correction to enforce discrete volume conservation. To ensure this, an additional inner condition defined by a boolean value, i.e.,

$$\text{conv}(p)^i = \begin{cases} true, & \text{res}_p < \text{tol}_{inn} \\ false, & \text{otherwise} \end{cases} \quad (3.49)$$

with a prescribed inner tolerance tol_{inn} and the pressure residual error norm res_p , is introduced for the inner loop. The complete algorithm of SAAMPLE is outlined in algorithm 4.

3.5 Interface capturing methods

As discussed in section 2, no jumps occur in physical properties such as density, viscosity, and pressure resulting from surface tension forces within the bulk of each fluid phase. This implies that

$$\begin{aligned} \rho_c &= \rho_f = \rho^\pm, \\ \mu_c &= \mu_f = \mu^\pm, \end{aligned} \quad (3.50)$$

$$(-(\mathbf{g} \cdot \mathbf{x})\nabla\rho + \sigma\kappa\nabla c)_c = 0$$

in the semi-discrete momentum equation eq. (3.33). However, these jumps must be accounted for in the interfacial region, necessitating an interface-capturing method prior to solving eq. (3.33). Over the past decades, numerous interface-capturing methods have been developed and extensively reviewed in the literature [18, 21, 33, 34].

Algorithm 4 Algorithm for a single SAAMPLE time step $[t^n, t^{n+1}]$

```
1:  $\text{conv}(F_f)^o := false$ 
2:  $\text{conv}(F_f)^i := false$ 
3:  $o := 0$ 
4: while  $o < N_{outer}$  and not  $\text{conv}(F_f)^o$  do
5:   if not  $\text{conv}(F_f)^i$  then
6:     Update mass flux:  $m_f := \rho_f F_f$ 
7:   end if
8:   Solve momentum prediction equation implicitly for  $\mathbf{v}^{*(o)}$  ▷ Equation (3.38)
9:    $i := 0$ 
10:  correct-pressure:= true
11:  while  $i < N_{inner}$  and correct-pressure do
12:    Set up pressure correction equation for  $p^{(i)}$  ▷ Equation (3.43)
13:    Update the  $\text{conv}(p)^i$  ▷ Equation (3.49)
14:    if not  $\text{conv}(p)^i$  then
15:      Update of  $\mathbf{v}^{(i)}$  and  $p^{(i)}$  with under-relaxation factors ▷ Equations (3.44) and (3.46)
16:      Update intermediate pressure and velocity ▷ Equation (3.47)
17:      Update volumetric fluxes  $F_f$ 
18:    else
19:      correct-pressure:= false
20:      if  $i = 0$  and  $\text{conv}(p)^i$  then
21:        Update fields  $\mathbf{v}^{o+1} = \mathbf{v}^{(i)}$ ,  $p^{o+1} = p^{(i)}$  for the next outer loop
22:         $\text{conv}(F_f)^o := true$ 
23:      end if
24:    end if
25:     $++i$ 
26:  end while
27:  $++o$ 
28: end while
29: Update fields  $\mathbf{v}^{n+1} = \mathbf{v}^{(o)}$ ,  $p^{n+1} = p^{(o)}$  for the next time step
30:
```

In this section, four methods are introduced. The fundamental concepts of the Front Tracking Method (FTM) and Level Set Method (LSM) are initially outlined in sections 3.5.1 and 3.5.2, respectively, laying the groundwork for the unstructured Level Set/Front Tracking (LENT) method [35], which is employed in section 3.5.4. Subsequently, the basic principles of the unstructured geometrical Volume of Fluid (VOF) method and detailed information on *isoAdvector* [36], the exact VOF method utilized in section 6.4 and section 9.6, are elucidated.

3.5.1 Front Tracking method

In FTM, the interface between two fluids is marked by a separate *front*, which consists of *marker points* connected by elements, which are line segments in two dimensions or triangles in three dimensions. These points evolve with the flow and then reconstruct the approximated phase indicator field from the location of the front. Figure 4 displays schematically the representation of an interface $\Sigma(t)$ approximated by a front $\tilde{\Sigma}(t)$ using connected marker points, e.g. the k th point \mathbf{x}_{Σ}^k , in two dimensions domain Ω .

Tryggvason et al. [21] provided a concise historical overview of FTM and outlined key considerations associated with this method:

- *Data structure for the front:* The front encompasses information about each marker point, including its location and index, as well as the connectivity of these points and a description of the physics at the interface. Effective management of this complex information requires an appropriate data structure, which significantly influences the performance of an FTM.
- *Front restructuring:* As the front undergoes motion, it undergoes deformation, stretching, and compression with the flow. These geometrical changes can result in a highly non-uniform distribution of front elements, potentially compromising the quality of the front. Therefore, dynamic evaluation and restructuring of the front during the simulation are necessary. Restructuring operations involve element manipulations such as addition, deletion, and reshaping [37], all of which depend critically on the flexibility of the data structure.

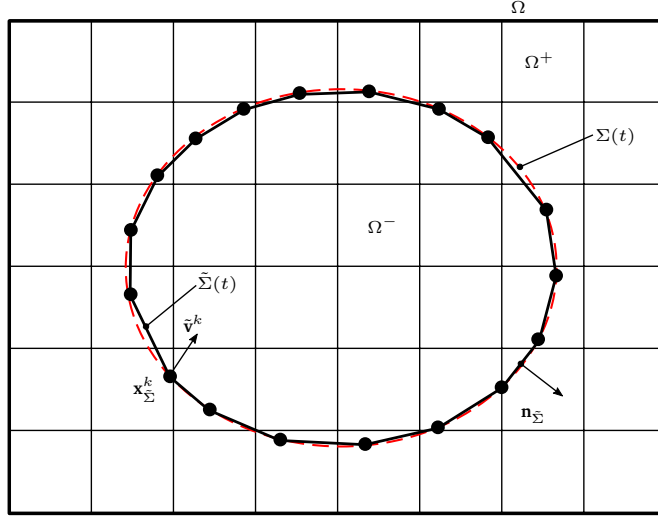


Figure 4: Exemplary illustration of interface approximation using FTM at time t : the red dashed curve indicates the exact interface $\Sigma(t)$, the black line segments connected by marker points (black dots) construct the front $\tilde{\Sigma}(t)$ with front normal $\mathbf{n}_{\tilde{\Sigma}(t)}$ of each segment, the k th marker points $\mathbf{x}_{\tilde{\Sigma}}^k$ moves with the interpolated velocity $\tilde{\mathbf{v}}^k$.

- *Communication between the front and fixed mesh cells:* The governing conservation equations are solved at fixed cell centers or cell faces of a mesh, while the marker points can move anywhere within the computational domain. Hence, establishing communication between the front and the governing equations is essential. The moving front requires information about the velocity field, discretely solved on the mesh, necessitating interpolation from mesh points to marker points. With the interpolated velocity on the front, the new position of a marker point, exemplified by k th marker point, is explicitly updated in the time interval $[t^n, t^{n+1}]$ by

$$\mathbf{x}_{\tilde{\Sigma}}^{k,n+1} = \mathbf{x}_{\tilde{\Sigma}}^{k,n} + \int_{t^n}^{t^{n+1}} \tilde{\mathbf{v}}(\mathbf{x}_{\tilde{\Sigma}}^k, t) dt, \quad (3.51)$$

where $\tilde{\mathbf{v}}(\mathbf{x}_{\tilde{\Sigma}}^k, t)$ indicates the interpolated velocity at the marker point. Conversely, transmitting information contained in the front to the mesh is more intricate. Theoretically, the phase indicator has a sharp jump across the front, leading to abrupt changes in

fluid properties from one mesh point to the next and resulting in numerical instability in solving the governing equations. To mitigate these issues and smooth the changes, the interface is given a small thickness on the order of the mesh size. A smoothing phase indicator function is generally defined by

$$\tilde{\chi}(\mathbf{x}) = \begin{cases} 1 & (\mathbf{x} - \mathbf{x}'_{\Sigma}) < -\epsilon h, \\ H(\mathbf{x}) & -\epsilon h \leq (\mathbf{x} - \mathbf{x}'_{\Sigma}) \leq \epsilon h, \\ 0 & (\mathbf{x} - \mathbf{x}'_{\Sigma}) > \epsilon h, \end{cases} \quad (3.52)$$

where \mathbf{x}'_{Σ} is the projected point from \mathbf{x} to the front in the normal direction \mathbf{n}_{Σ} , as depicted in fig. 4, h is the cell size, ϵ is an empirical parameter to control the smoothing thickness and $H(\mathbf{x})$ is the smoothing function in the interfacial region and varies in different methods, e.g. [4, 5, 37, 38]. The density and viscosity can be represented by eqs. (2.10) and (2.11) that are replaced with the smoothed phase indicator $\tilde{\chi}$. The surface tension is approximated by CSF model, i.e., eq. (3.24) with $c(\mathbf{x}) = \tilde{\chi}$ and $\kappa = -\nabla \cdot \tilde{\mathbf{n}}$, where the $\tilde{\mathbf{n}}$ is the normal interpolated from the normals of the near front elements. Besides the smoothing methods, some methods attempt to maintain the sharpness [39, 40]. One option is Ghost Fluid Method (GFM) [41, 42], where the "ghost" values across the front are extrapolated from another side of the front and the mesh is still fixed.

3.5.2 Level Set method

The LSM uses the level set function $\phi(\mathbf{x}, t)$ to represent the interface between two phases. Traditionally, this function is a signed distance function[43, 44], representing the shortest distance to the interface, defined as follows:

$$\phi(\mathbf{x}, t) = \begin{cases} -dist(\mathbf{x}(t), \Sigma(t)) & \mathbf{x}(t) \in \Omega^-, \\ +dist(\mathbf{x}(t), \Sigma(t)) & \mathbf{x}(t) \in \Omega^+, \\ 0 & \mathbf{x}(t) \in \Sigma(t), \end{cases} \quad (3.53)$$

where $dist(\mathbf{x}(t), \Sigma(t))$ is the Euclidian distance to the interface $\Sigma(t) := \{\mathbf{x}, \phi(\mathbf{x}, t) = 0\}$, i.e., constructed by the points at the level curve $\phi(\mathbf{x}, t) = 0$. The level set function evolves with the flow, following the advection equation

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \mathbf{v}) = 0, \quad (3.54)$$

where \mathbf{v} is the flow velocity, in an incompressible setting $\nabla \cdot \mathbf{v} = 0$. The discretization schemes for time and space follow the discussion in section 3.3.5 and section 3.3.2. The similar smoothed phase indicator function to eq. (3.52) is employed to reconstruct density and viscosity, taking the form as

$$\tilde{\chi}(\mathbf{x}) = \begin{cases} 1 & \phi(\mathbf{x}) < -\epsilon h, \\ H(\mathbf{x}) & -\epsilon h \leq \phi(\mathbf{x}) \leq \epsilon h, \\ 0 & \phi(\mathbf{x}) > \epsilon h, \end{cases} \quad (3.55)$$

where the smoothing function $H(\mathbf{x})$ can be selected same as in eq. (3.52). The level set

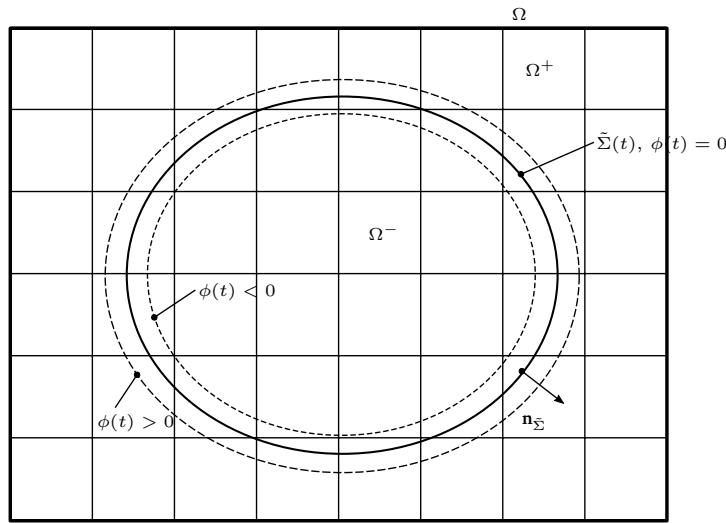


Figure 5: Exemplary illustration of interface approximation using LSM at time t .

function can also be used to compute the interface normal \mathbf{n}_{Σ} , with the characteristic function

$c(\mathbf{x}) = \phi(\mathbf{x})$ substituted in general CSF model for surface tension eq. (3.24), as

$$\begin{aligned}\mathbf{n}_\Sigma &= \frac{\nabla\phi}{|\nabla\phi|}, \\ \mathbf{f}_\Sigma &= \sigma\kappa\nabla\tilde{\chi}, \text{ with } \kappa = -\nabla \cdot \mathbf{n}_\Sigma.\end{aligned}\tag{3.56}$$

Since the level set function does not retain its signed distance function property $|\nabla\phi| = 1$ as it evolves in time, it is necessary to *reinitialize* the level set function periodically. Without reinitialization, the magnitude of the level set function gradient $|\nabla\phi|$ can become very large or small near the interface, i.e. $\phi = 0$, leading to numerical instability and a loss of accuracy. Sussman, Smereka, and Osher [44] proposed an iterative approach to reinitialize ϕ by solving the following equation

$$\frac{\partial\phi}{\partial\tau} = \text{sgn}(\phi^{\tau=0})(|\nabla\phi| - 1)\tag{3.57}$$

where the τ is an artificial time, $\phi^{\tau=0}$ denotes the un-reinitialized level set function calculated from eq. (3.54) and sgn indicates a smoothed signum function with definition as following

$$\text{sgn}(\phi) = \begin{cases} -1 & \phi < 0, \\ 0 & \phi = 0, \\ 1 & \phi > 0, \end{cases}\tag{3.58}$$

After several iterations, the adjusted ϕ stabilizes, ensuring $|\nabla\phi| = 1$ and thereby reinstating the signed distance property of ϕ . However, reinitialization introduces a new issue: the artificial displacement of the interface ($\phi = 0$) violates mass conservation. Addressing this challenge necessitates additional measures, further complicating the LSM [45].

3.5.3 Hybrid Level Set / Front Tracking method (LENT)

The LENT[38, 46, 47] emerges as a hybrid approach, amalgamating the strengths of FTM and LSM while operating on unstructured meshes. Similar to FTM discussed in section 3.5.1, this method represents the interface via the front, evolving alongside marker points along discrete Lagrangian trajectories. Additionally, akin to LSM, it computes the signed distance field in the immediate proximity of the front cells. This field aids in restructuring the front through the application of an iso-surface reconstruction algorithm[38, 48].

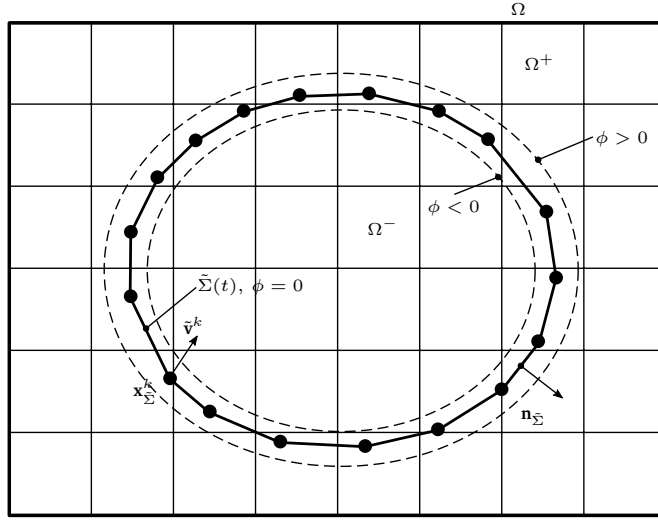


Figure 6: Exemplary illustration of interface approximation using LENT at time t .

Interface evolution The marker points move following the classical FTM as described by eq. (3.51) with the interpolated velocity from neighboring cell centers. The neighboring cells relative to each marker point must be specified before interpolating. Unlike in structured meshes where cell indices suffice for locating, unstructured meshes require additional more complex steps to determine the neighboring cells of a marker point. This is accomplished through a fusion of octree space subdivision and known-neighborhood search algorithm[35].

Signed distance calculation In contrast to conventional level set methods, the evolution of the signed distance field ϕ does not rely on solving an advection equation eq. (3.54). Instead, ϕ is geometrically computed from the front $\tilde{\Sigma}$ at each time step. For a given point \mathbf{x} , $\phi(\mathbf{x})$ is determined as follows

$$\phi(\mathbf{x}) = \text{sgn}((\mathbf{x} - \mathbf{x}_{\tilde{\Sigma}, \text{closest}}) \cdot \mathbf{n}_{\mathcal{T}}) |\mathbf{x} - \mathbf{x}_{\tilde{\Sigma}, \text{closest}}|, \quad (3.59)$$

where $\mathbf{x}_{\tilde{\Sigma}, \text{closest}}$ represents the point on the interface $\tilde{\Sigma}$ closest to \mathbf{x} . The sign is determined with the normal vector $\mathbf{n}_{\mathcal{T}}$ of the triangle containing $\mathbf{x}_{\tilde{\Sigma}, \text{closest}}$. Given that ϕ is primarily needed in the vicinity of $\tilde{\Sigma}$, octree space subdivision is employed to confine computations to

a narrow band. A comprehensive explanation of this process is provided in [35]. The same approximation as eq. (3.56) is used to evaluate surface tension force.

3.5.4 Volume of Fluid method

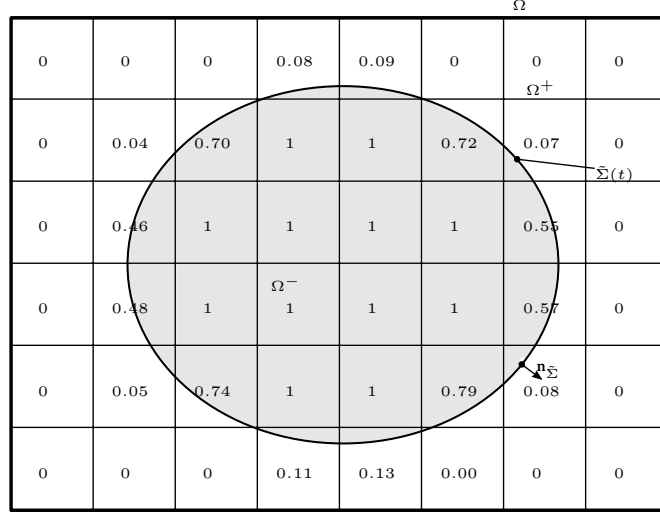


Figure 7: Exemplary illustration of interface approximation using VOF at time t with marked volume fraction in each cell.

In VOF, the phase interface is represented by the volume fraction of a phase liquid in each cell. The volume fraction is the discrete version of the phase indicator function (eq. (2.9)) and is defined in each cell, e.g. for the cell Ω_c , as

$$\alpha_c(t) := \frac{1}{|\Omega_c|} \int_{\Omega_c} \chi(\mathbf{x}, t) dV. \quad (3.60)$$

Accordingly, the one-field density and viscosity are calculated cell-wisely as

$$\rho_c(t) = \rho^- \alpha_c(t) + \rho^+ (1 - \alpha_c(t)), \quad (3.61)$$

$$\mu_c(t) = \mu^- \alpha_c(t) + \mu^+ (1 - \alpha_c(t)). \quad (3.62)$$

in the cell Ω_c , corresponding to eq. (2.10) and eq. (2.11). Integrating the conservative mass equation eq. (2.1) over a control volume Ω_c yields

$$\frac{\partial}{\partial t} \int_{\Omega_c} \rho dV + \int_{\partial\Omega_c} \rho \mathbf{v} \cdot d\mathbf{S}_f = 0. \quad (3.63)$$

Applying one-field density eq. (2.10) to eq. (3.63) results in

$$\frac{\partial}{\partial t} \int_{\Omega_c} [(\rho^- - \rho^+) \chi + \rho^+] dV + \int_{\partial\Omega_c} [(\rho^- - \rho^+) \chi + \rho^+] \mathbf{v} \cdot d\mathbf{S}_f = 0. \quad (3.64)$$

Equation (3.64) is reformulated as following

$$(\rho^- - \rho^+) \frac{\partial}{\partial t} \int_{\Omega_c} \chi dV + (\rho^- - \rho^+) \int_{\partial\Omega_c} \chi \mathbf{v} \cdot \mathbf{n} dS + \rho^+ \int_{\partial\Omega_c} \mathbf{v} \cdot d\mathbf{S}_f = 0. \quad (3.65)$$

Integrating continuity equation $\nabla \cdot \mathbf{v} = 0$ in a same way gives

$$\int_{\Omega_c} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega_c} \mathbf{v} \cdot d\mathbf{S}_f = 0. \quad (3.66)$$

The integral form of phase indicator transport equation is acquired by inserting eq. (3.66) to eq. (3.65) and dividing the constant $(\rho^- - \rho^+)$, i.e.,

$$\frac{\partial}{\partial t} \int_{\Omega_c} \chi dV + \int_{\partial\Omega_c} \chi \mathbf{v} \cdot d\mathbf{S}_f = 0. \quad (3.67)$$

Applying the volume average to eq. (3.67) results in

$$\frac{\partial}{\partial t} \frac{1}{|\Omega_c|} \int_{\Omega_c} \chi dV + \frac{1}{|\Omega_c|} \int_{\partial\Omega_c} \chi \mathbf{v} \cdot d\mathbf{S}_f = 0. \quad (3.68)$$

The conservative transport of volume fraction within Ω_c is obtained by substituting eq. (3.60) into eq. (3.68), i.e.,

$$\frac{\partial \alpha_c}{\partial t} + \frac{1}{|\Omega_c|} \int_{\partial\Omega_c} \chi \mathbf{v} \cdot d\mathbf{S}_f = 0. \quad (3.69)$$

Integrating eq. (3.69) from t^n to t^{n+1} gives the following equation for the new volume fraction of cell Ω_c ,

$$\alpha_c^{n+1} = \alpha_c^n - \frac{1}{|\Omega_c|} \sum_{f \in F_c} s_f \int_{t^n}^{t^{n+1}} \int_f \chi \mathbf{v} \cdot d\mathbf{S}_f dt = \alpha_c^n - \frac{1}{|\Omega_c|} \sum_{f \in F_c} s_f V_f^\alpha, \quad (3.70)$$

where s_f represents the sign of the volumetric flux, i.e. $s_f = \text{sgn}(F_f)$, which is used to ensure that $s_f d\mathbf{S}_f$ always points out of the cell, and V_f^α is named the *phase-specific volume*, defined as

$$V_f^\alpha = \int_{t^n}^{t^{n+1}} \int_{S_f} \chi \mathbf{v} \cdot d\mathbf{S}_f dt. \quad (3.71)$$

The VOF methods can be divided into two main categories based on the approach to transport and approximate the volume fraction α : *algebraic* VOF methods and *geometric* VOF methods[33]. Algebraic VOF methods represent α using a polynomial or hyperbolic-tangent function. These methods algebraically compute the phase-specific volume without the need for geometric interface reconstruction, hence their name. Conversely, geometric VOF methods involve reconstructing the phase indicator function χ using a cellwise continuous geometrical approximation and eq. (3.60). Subsequently, the reconstructed interface is advected by calculating the phase-specific volume across each interfacial cell using geometric techniques. The volume fraction α is employed as the characteristic function in CSF model (eq. (3.24)) to estimate the surface tension in this study.

In section 6.4 and section 9.6, a geometric VOF method named *isoAdvector* [6, 36, 49] is adopted to capture the interface. A brief review of the *isoAdvector* method is outlined below.

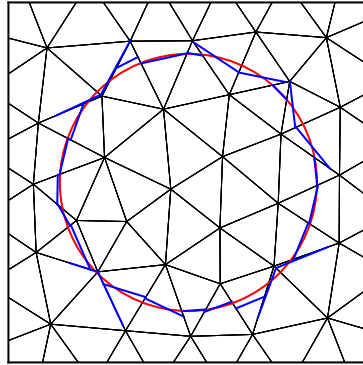


Figure 8: PLIC interface approximated by Youngs' algorithm [1] (blue lines) and exact interface (red circle).

Interface Reconstruction With only volume fraction information available and no actual interface data, specifying the distribution of two liquids in an interfacial cell becomes challenging, rendering geometric calculation of phase-specific volume difficult. To surmount this obstacle, a specialized technique for interface reconstruction is essential. The interface is characterized by its center positions \mathbf{x}_Σ and area normals \mathbf{n}_Σ .

In the original isoAdvect method [36], a α -isosurface based approach is employed. In this method, cell volume fraction data is interpolated to the cell vertices. An initial guessed isovalue, α_0 , is assigned to construct the α_0 -isosurface inside the cell by examining all the cell's edges and determining whether they are intersected by the isosurface. An edge is considered intersected if the interpolated volume fraction at one end is greater than α_0 and the value at the other end is smaller than α_0 . If this condition is met, the intersection point along the edge is calculated via linear interpolation. Connecting these intersection points across the cell faces enables construction of the isosurface in the interfacial cell. Since α_0 is initialized with a guessed value, the sub-volume of the intersected cell typically does not correspond to the volume fraction. To determine the isovalue capable of recovering the volume fraction, a sweeping process is required. The same computational method for the interface centers \mathbf{x}_Σ and area normals \mathbf{n}_Σ is utilized as described in eq. (3.14) and eq. (3.13).

In the subsequent study by Scheufler and Roenby [6], two additional schemes based on Reconstructed Distance Function (RDF)-based schemes are introduced for the computational interface reconstruction. The RDF is computed as a weighted average of distances to the interfaces within the cell itself and neighboring cell layers composed of point neighbors. While similar to a signed distance function discussed in section 3.5.2, the RDF does not necessarily constrain the magnitude of the gradient of the signed distance function to be 1. In the first scheme, iso-RDF is employed instead of iso-alpha value to construct the iso-surface. Differing from iso-approaches, where interface centers and normals are determined from intersection points of the iso-surface, the interface normal \mathbf{n}_Σ is algebraically obtained by initially computing the unit gradient of the RDF. The conventional Piecewise Linear Interface Calculation (PLIC) method is employed to capture the interface centers. In PLIC, the interface is approximated in each interfacial cell as a line in two dimensions and a plane in three dimensions given by

$$\mathbf{n}_\Sigma \cdot \mathbf{x}_\Sigma + C = 0, \quad (3.72)$$

where C is the constant in the plane equation. In this scheme, C is computed through sweeping to ensure that the volume cut by the interface Σ equals the volume fraction in the computational cell to ensure eq. (3.60) is satisfied. The calculation of RDF and interface reconstruction are performed iteratively in both RDF-based schemes, guaranteeing second-order convergence of interface normals. The combined scheme is named plicRDF.

In this study, the plicRDF scheme is employed to reconstruct the interface.

Interface Advection Given the known interface normals and centers, the phase-specific volume V_f^α can be estimated. To decouple eq. (3.70) from the momentum transport equation, the velocity field in eq. (3.70) is assumed to remain linear, resulting in constant volumetric fluxes F_f across mesh faces. The term $\mathbf{v} \cdot d\mathbf{S}$ in eq. (3.71) can be approximated by

$$\mathbf{v} \cdot d\mathbf{S} \approx \frac{F_f}{|\mathbf{S}_f|} dS, \quad (3.73)$$

with second order accuracy in space, where $|\mathbf{S}_f|$ is the area of cell face f , dS is the differential of face area of f . Substituting this approximation into eq. (3.71) yields

$$V_f^\alpha \approx \frac{0.5(F_f^n + F_f^{n+1})}{|\mathbf{S}_f|} \int_{t^n}^{t^{n+1}} \int_f \chi dS dt. \quad (3.74)$$

The area of the cell face f submerged in the Ω^- is defined as

$$A_f = \int_f \chi dS. \quad (3.75)$$

The phase-specific volume is then approximated as

$$V_f^\alpha \approx \frac{0.5(F_f^n + F_f^{n+1})}{|\mathbf{S}_f|} \int_{t^n}^{t^{n+1}} A_f dt. \quad (3.76)$$

The only remaining term to estimate is the time integral of A_f within the time step $[t^n, t^{n+1}]$. Roenby, Bredmose, and Jasak [36] devised a discontinuous function of t for the submerged area A_f , derived from iso-surface motion and the submerged polygonal face calculation. To ensure that the updated volume fraction remains within a physically feasible range, i.e., $\alpha^{n+1} \in [0, 1]$, a bounding procedure is conducted. This procedure iteratively redistributes any undershoot or overshoot of volume in a cell to neighboring cells.

Hybrid Level Set / Front Tracking method with high density-ratios

4 Introduction

A variety of natural and industrial two-phase flow processes involve gas/liquid flows, characterized by density ratios $\rho^-/\rho^+ \geq 10^3$, such as the atomization of fuel jets [8], sloshing tank [9], mold filling [10], water flooding [11]. Large density ratios at the fluid interface cause severe challenges for numerical simulations [50]. For segregated solvers, the discrete pressure Poisson equation becomes ill-conditioned if density is cell-centered, since its abrupt change across the interface between the two fluids can lead to a large variation in the matrix coefficients. Additionally, spurious numerical errors in the solution of the momentum equation accumulate because of inconsistencies between mass and momentum advection.

Ghods and Herrmann [51] point out that for level set methods mass and momentum are typically transported in different, inconsistent ways. While mass is transported by a solution of the level set equation, momentum is obtained from solving a non-conservative form of a momentum balance equation. Hence, a large non-physical change in the momentum can be generated by a small error in the interface position when the density ratio is high. Nangia et al. [52] state that the abrupt change in density often introduces notable shear at the interface and adds difficulties in the discretization of governing momentum equations at the interface, which further leads to higher stiffness of the linear equation system.

Many researchers have addressed these problems, and some indicated further that because of the sizeable numerical error resulting from high-density ratios, some flow algorithms

or solvers can only be used to solve low density-ratio cases with $\rho^-/\rho^+ \in [1, 10]$ [53]. However, in engineering applications, density ratios usually range from 10^2 to 10^3 , and even 10^4 for molten metals or water-water vapor systems. Hence, a solution algorithm with the ability to handle a broader range of density ratio problems is required to simulate real-world engineering problems.

In this chapter, the unstructured LENT [2, 35] is extended for handling two-phase flows with strongly different densities (high-density ratios) by providing the theoretical basis for the numerical consistency between the mass and momentum conservation in the collocated Finite Volume discretization of the single-field two-phase Navier-Stokes equations. The analysis provides the theoretical basis for the mass conservation equation introduced by Ghods and Herrmann [51] and used in [7, 52–55].

A mass flux that is consistent with mass conservation in the implicit Finite Volume discretization of the two-phase momentum convection term, and solve the single-field Navier-Stokes equations with SAAMPLE segregated solution algorithm [2], is used in this study. The proposed ρ LENT method recovers exact numerical stability for the two-phase momentum advection of a spherical droplet with density ratios $\rho^-/\rho^+ \in [1, 10^4]$. Numerical stability is demonstrated for in terms of the relative L_∞ velocity error norm, for density-ratios in the range of $[1, 10^4]$, dynamic viscosity-ratios in the range of $[1, 10^4]$ and very strong surface tension forces, for challenging mercury/air and water/air fluid pairings. In addition, the solver performs well in cases characterized by strong interaction between two phases, i.e., oscillating droplets and rising bubbles.

The proposed ρ LENT method¹ is applicable to any other two-phase flow simulation method that discretizes the single-field two-phase Navier-Stokes Equations using the collocated unstructured Finite Volume Method but does not solve an advection equation for the phase indicator using a flux-based approach, by adding the proposed geometrical approximation of the mass flux and the auxiliary mass conservation equation to the solution algorithm.

¹The implementation in OpenFOAM is publicly available at <https://gitlab.com/leia-methods/lent/-/tree/2022-02-rhoLENT-R1> [56].

Literature review

A pioneering attempt to alleviate numerical instability of the VOF method caused by high-density ratios was made by Rudman [57]. Rudman [57] has used a sub-mesh with a doubled mesh resolution for advecting volume fractions, compared to the mesh used for the momentum and pressure equations. The goal of this two mesh approach was the reduction of small errors in the discrete momentum that cause large errors in the velocity. However, an additional higher mesh resolution for the volume fractions requires a discrete divergence free velocity on the finer mesh. Furthermore, using an additional mesh for the volume fractions increases the computational costs significantly, and it is not applicable to general unstructured meshes. Rudman [57] demonstrates qualitatively a reduction of parasitic currents for the stationary droplet case with $\rho^-/\rho^+ = 100$, and improved results for more complex cases. Another important finding of Rudman [57] is the role of the densities used in the mass flux and the momentum flux in ensuring numerical consistency of the two-phase momentum advection.

Bussmann, Kothe, and Sicilian [58] extended the work of Rudman [57] for the unstructured collocated finite volume method. Bussmann, Kothe, and Sicilian [58] employ the conservative form for the momentum convection. At first, the momentum advection is solved separately, using an explicit Euler time integration scheme. Bussmann, Kothe, and Sicilian [58] use the unstructured unsplit Volume-of-Fluid method of Rider and Kothe [59], which enables the simplification of the numerical consistency requirement for the density and momentum equations. Specifically, the solution of the volume fraction equation results in phase-specific volumes at face centers. Those phase-specific volumes are then used to compute the volume fractions at face centers. These volume fractions are used by Bussmann, Kothe, and Sicilian [58], together with a simple average of cell densities, and velocities calculated by the least squares reconstruction technique, to compute the momentum fluxes at face centers. Since the velocity is continuous at the interface, the least squares approximation is acceptable. However, calculating face-centered densities by an average does not yield numerical stability in all cases. Contrary to Rudman [57], Bussmann, Kothe, and Sicilian [58] do not require an additional finer mesh. They do, however, limit the solution to first-order accuracy in time and introduce the Courant-Friedrichs-Lewy (CFL) condition by solving the momentum advection equation explicitly. Bussmann, Kothe, and Sicilian [58] introduce the important case of a

translating droplet in a quiescent ambient fluid. This test case can be used to demonstrate numerical consistency in the momentum transport. Their solutions show accurate results for high density ratios, especially considering the fact that even the unsplit VOF method distorts the interface during the translation [60]. However, for $\rho^-/\rho^+ \in [1, 10^2]$, the constant translation velocity is modified by the solution of the pressure and momentum equations, which implies a remaining numerical inconsistency in this approach.

Sussman et al. [61] employ the Coupled Level Set and Volume Fluid Method (CLSVOF) method [62] for obtaining a robust and stable solution for the density ratio of 1000 by extrapolating the liquid velocities into the gas domain. The interface is advected using the extrapolated liquid velocity field only.

Raessi and Pitsch [53] propose a 2D staggered discretization of conservative single-field form of two-phase Navier-Stokes equations for handling high density ratios. Like Bussmann, Kothe, and Sicilian [58] did, Raessi and Pitsch [53] first solve the momentum advection equation, using second-order (or higher) explicit integration schemes, and upwinding for the velocity near the interface. The density used in the momentum convective term is computed as a weighted combination of signed distances from the old and the new time step. For the partially submerged line segments bounding 2D rectangular cells, intersection between the mesh and the zero level set (iso-surface) is performed using the marching cubes algorithm. Raessi and Pitsch [53] point out that there is still an inconsistency between the face-centered density and the momentum transport, as the Level Set equation remains decoupled / inconsistent with the momentum transport. The verification of numerical stability was done using the translating droplet case from Bussmann, Kothe, and Sicilian [58], and results demonstrate qualitative improvement for the density ratio $\rho^-/\rho^+ = 10^6$. Other density ratios have not been verified. A viscous oscillating droplet case demonstrates quantitative improvement in terms of the improved amplitude decay rate, compared to non-conservative form of the momentum equation.

Le Chenadec and Pitsch [63] extend their forward/backward Lagrangian tracking and Eulerian remapping VOF method [63] for handling high density ratios. Equivalent to volume fractions in [63], the density and the momentum are advected in the Lagrangian forward/backward tracking step by observing the control volume as a material volume and moving the mesh forward / backward with the flow velocity. While the content of material volumes

does not change on the continuum level, this condition cannot be discretely ensured and is a source of conservation errors. In the Eulerian re-mapping step, physical properties are transferred from the Lagrangian to the Eulerian mesh, and the geometrical intersections between the PLIC interface on the forward/backward image of the mesh, and the background mesh, are another source of volume conservation errors. Ensuring numerical consistency further requires the transfer of velocities located at the center of mass. Since the velocities associated with the cell centroids are used, an inconsistency is introduced. Qualitative results show significant improvements for the stationary droplet with $\rho^-/\rho^+ = 10^9$, and quantitative improvement is shown for the standing wave by Prosperetti [64] with $\rho^-/\rho^+ = 850$.

Ghods and Herrmann [51] have developed a Consistent Rescaled Momentum Transport (CRMT) method. The CRMT method discretizes the conservative form of the single-field Navier-Stokes equations using a collocated unstructured Finite Volume method. To increase the numerical stability for high density ratio, CRMT solves an "auxiliary" mass conservation equation using a mass flux either by upwinding the face-centered density in the interface cells and their face-neighbors (defined by a volume fraction tolerance), or by averaging the densities elsewhere. The same discretization scheme used for the face-centered density is also applied to the mass flux in the convective term of the momentum equation. A difference is therefore introduced in the mass flux of the continuity equation and the mass flux in the convective term of the momentum equation when upwinding is used, because the upwinded face-centered density in the continuity equation uses the face-centered velocity, while the upwinded mass flux in the momentum equation includes both the upwind velocity and density. In this chapter, it is shown that any difference in the discretization of the mass flux to be a source of numerical inconsistency for the two-phase momentum advection. Like Bussmann, Kothe, and Sicilian [58], the explicit discretization of the momentum convective term introduces the CFL condition, limiting the time step for convection-dominated multiphase flows, where high density ratios play a major role. Using upwind schemes makes the discretization first-order accurate. The droplet translation case [58], with $\rho^-/\rho^+ = 10^6$, is compared in terms of the droplet shape, that remains stable. Other density ratios are not reported for this verification case. It is considered, that the droplet shape errors may result from the interface

advection scheme², and should be generally substituted by the L_∞ norm of the velocity error to demonstrate numerical consistency.

Vaudor et al. [65] base their approach on a CLSVOF code from Ménard, Tanguy, and Berlemont [66] and Aniszewski, Ménard, and Marek [67], which can switch between LSM-based and VOF-based mode to calculate momentum fluxes. They [65] chose the VOF-based momentum fluxes calculation mode and implemented the framework of Rudman's method [57] but with more accurate interpolation schemes for velocities and velocity gradients on faces of staggered meshes to ensure consistency. This method is developed in two-dimensions and exploits two sets of meshes. To provide a more widely applicable method, Vaudor et al. [68] advanced the method in their more recent study. In contrast to the previous work [65], the LS method tracks the interface, while the VOF method is utilized to update density. They exploited the identical scheme to discretize conservative convective term in mass and momentum equation. In addition, the mass flux is also identical in both discretized equations. A new strategy that leverages half cell-faces' and half cells' quantities of volume fraction and density to couple staggered mass cells and momentum cells is introduced to avoid the need for a refined mesh in the original method by Rudman [57]. A prominent feature of this new method is that it can be used to simulate three-dimensional applications. Besides, comparing with the method from Rudman [57], the new method shows relatively low computational cost when simulating the same 2D application.

Owkes and Desjardins [69] presented a three-dimensional, unsplit, second-order semi-Lagrangian VOF scheme that conserves mass and momentum and ensures consistency between the mass (volume fraction) and momentum fluxes. The volume fractions are geometrically transported near the fluid interface using the method from [70]. As in [57], Owkes and Desjardins [69] introduce an additional refined mesh for the calculation of semi-Lagrangian fluxes. The motivation for the refined mesh is to enforce the consistency between semi-Lagrangian mass and momentum fluxes, similar to Rudman [57]. Results confirm mass and momentum conservation, and stability of the momentum convection. The method proposed by Owkes and Desjardins [69] relies on the staggered variable arrangement and this, together with the use of the additional finer mesh, makes this approach inapplicable to unstructured

²The Level Set and VOF methods do not exactly preserve the shape of a translating droplet.

finite volume meshes.

Orazzo et al. [54], similarly to Rudman [57], resolve the volume fraction function on twice finer sub-cells and update density from the volume fraction. After that, they update face-centered density on mass cells by averaging density on sub-cells, and then evaluate the mass flux on the faces of standard staggered momentum cells. These density and mass flux values are used to initialize and calculate interim momentum and velocity during the prediction step. Zuzio et al. [7] made no changes and applied Orazzo’s method [54]. Besides, they further verified and validated this method with more complex cases, e.g., liquid jet in cross-flow. Yang, Lu, and Wang [71] notice that the high-density ratio has a profound effect on robustly simulating two-phase flows at high Reynolds numbers. To mitigate the problem, they adopt the consistent framework from Nangia et al. [52] and replace the interface-capturing method in [52], which is standard LSM, with CLSVOF method [62] to ensure mass conservation.

Patel and Natarajan [72] employ the method of Ghods and Herrmann [51], a high-resolution scheme called Cubic Upwind Interpolation (CUI) for the convective terms of momentum and volume fraction transport equations, and the solution of a momentum equation in the face-normal direction. The face-normal momentum equation leads to a combined collocated/staggered variable arrangement, that requires the use of nonlinear solvers, as this equation is a non-linear algebraic equation. Patel and Natarajan [72] demonstrate the balanced nature of their discretization for the stationary droplet using exact curvature and density ratios $\rho^-/\rho^+ \in [10, 1000]$. Numerical stability is demonstrated with reduced parasitic currents when the curvature is approximated numerically for $\rho^-/\rho^+ = 10$, $We = 1$. For the verification test case of the two-phase momentum advection problem, $\rho^-/\rho^+ = 10^6$ is used without surface tension and viscous forces and qualitative results show slight deformations of the interface shape, the L_∞ norm of the velocity error is not reported. With enabled surface tension and viscous forces and exact curvature prescribed, and density ratios $\rho^-/\rho^+ = 1, 1000$, the velocity error in the L_∞ norm lies within $[10^{-3}, 10^{-2}]$.

Manik, Dalal, and Natarajan [73], similarly to [72], attempt to enforce numerical consistency by applying the similar discretization scheme on the conservative form of the volume fraction advection equation and the momentum conservation equation. Manik, Dalal, and Natarajan [73] are using a collocated unstructured Finite Volume method for the equation discretization and the Convergent and Universally Bounded Interpolation Scheme for the

Treatment of Advection (CUBISTA) scheme (Alves, Oliveira, and Pinho [74]) to discretize convective terms. The verification of the numerical consistency for the two-phase momentum advection is done using the droplet translation case of Bussmann, Kothe, and Sicilian [58] and density ratios $\rho^-/\rho^+ = 10^3, 10^6$, that demonstrates qualitative improvement compared to a naive discretization of the momentum convective term with the upwind method. The qualitative evaluation is based on the shape of the droplet, given by the 0.5 iso-surface of the volume fraction. Although the proposed method demonstrates improvement w.r.t. an obviously inconsistent approach, some shape deformation is still visible, so one can conclude that $L_\infty(\mathbf{v}) \neq 0$ and some non-zero velocities are still generated.

A recent second-order accurate LSM is proposed by Nangia et al. [52], extending the work from Ghods and Herrmann [51] that is first-order accurate. Similar to the method proposed by Ghods and Herrmann [51], an additional mass conservation equation is solved, and the identical mass flux is used for both mass and momentum transport. Two techniques are employed: one is the third-order accurate Koren's limited CUI, which is modified to consistently discretize the convective term of both mass and momentum equation. This scheme satisfies the Convection-Boundedness Criterion (CBC) and is Total Variation Diminishing (TVD). The second technique is the solution of an update equation for the face-centred densities. In this step, a 3-order accurate Strong Stability Preserving Runge-Kutta (SSP-RK3) scheme is used for time integration. The update is performed in every fix-point iteration, and the updated face-centered density is then employed to solve the discretized momentum equation.

Zuzio et al. [7] also follow Ghods and Herrmann [51] by solving an auxiliary continuity equation for increasing the numerical consistency in discretizing the two-phase momentum convection term. Their Consistent Momentum-Mass (CMOM) transport method utilizes a staggered Cartesian variable arrangement and utilizes the two-phase incompressible Navier-Stokes equations in the conservative form, solved using Chorin's projection method [75] together with the CLSVOF method for tracking the fluid interface. The solution of the auxiliary density equation requires the evaluation of staggered (face-centered) densities, by constructing staggered control volumes, and evaluating the densities using sub-grid quadtree (octree in 3D) refinement and intersection with the PLIC interfaces. This aspect of CMOM shows the importance of evaluating the densities at face-centers that are required for the

solution of the auxiliary continuity equation. Momentum flux reconstruction scales the fluxed phase-specific volume from the VOF method. Finally, the two-phase momentum is advected in the staggered cells, and scaled with the corresponding density to obtain velocity components in all spatial directions. Zuzio et al. [7] demonstrate significant improvements in numerical stability in a very detailed way, reporting shape, position and kinetic energy errors for canonical verification and validation cases. The kinetic energy for the dense translating droplet [58] with a density ratio of 10^6 is reported, and CMOM recovers a numerically stable solution.

Arrufat et al. [76] consider the conservative form of the advection equation of a discontinuous property to enforce numerical consistency of the advected two-phase momentum, using face averages that are derived by integrating the advection equation in space and time. Since the discontinuity of the property introduced by the interface complicates the evaluation of the face averages, two additional equations are introduced, one for each phase. The method is derived for the Marker-And-Cell (MAC) staggered variable arrangement. Results demonstrate a numerically stable droplet shape when it is advected with a constant velocity, however, the authors consider this case to only test the consistency of the implementation and not the numerical consistency of the method - it is verified that it is important for both in the following sections - so the results are not quantified in terms of kinetic energy or L_∞ velocity errors. Still, the method shows significant improvements for realistic multiphase flows with high density ratios.

The high-density ratio is also challenging for other numerical methods for two-phase flows, like the phase-field and lattice Boltzmann. The corresponding surveys are beyond the scope of this work, more details can be found in [77–82]. Contrary to the numerical two-phase methods mentioned so far, the difficulties with high density ratios are far less pronounced for Front Tracking methods [83] because the marker field (phase-indicator) is not as sharp as in the unstructured Volume-of-Fluid method [34] and the unstructured Level Set / Front Tracking method [2, 35].

The methods of Ghods and Herrmann [51], Bussmann, Kothe, and Sicilian [58], Patel and Natarajan [72], and Manik, Dalal, and Natarajan [73] utilize the unstructured Finite Volume equation discretization, other above-mentioned methods utilize a staggered variable arrangement that is not applicable to unstructured meshes. Compared to contemporary

collocated Finite Volume methods, ρ LENT method proposed in this chapter achieves the numerical consistency in the two-phase momentum advection exactly. The requirement for the auxiliary mass conservation equation introduced by Ghods and Herrmann [51], and the requirement for the face-centered (flux) density from the mass conservation principle are derived in this chapter. Compared to a similar observation by [76], the integration in time that complicates the evaluation of face-centered quantities is avoided, as demonstrated in detail below. Although hybrid Level Set / Front Tracking LENT method [35] is used for interface capturing, the ρ LENT solution algorithm can be used with other interface capturing methods, where there is a discrepancy in the evaluation of the collocated density.

A collocated unstructured Finite Volume discretization is used in this study because it is ideal for geometrically complex domains. At its core, the proposed unstructured collocated finite-volume ρ LENT geometrically approximates the face-centered density in the mass flux and implicitly discretizes the two-phase momentum convective term, thus avoiding the interpolation of face-centered densities and the CFL stability criterion introduced in [58].

5 Methodology review

Hybrid multiphase flow simulation methods combine the sub-algorithms of FTM, LSM, or VOF to achieve better overall results. The structured Hybrid Level Set / Front Tracking method ([38, 46, 48, 83, 84]) has demonstrated remarkable capabilities for simulating a wide range of multiphase flows. The unstructured LENT method - as exhibited in [2, 35, 85] - shows promising computational efficiency and accuracy for surface tension driven flows on unstructured meshes.

However, LENT method in its existing form cannot handle two-phase flows with strongly different densities. Cell-centered volume fractions $\alpha_c(t)$ are computed from signed distances $\psi_c(t)$, that are computed geometrically from the Front $\tilde{\Sigma}(t) \approx \Sigma(t)$. The approximation algorithm for $\alpha_c(t)$ from $\psi_c(t)$ is detailed in [85]. This geometrical calculation of α_c from $\tilde{\Sigma}(t)$ and, subsequently, the calculation of $\rho_c(t)$ from $\alpha_c(t)$ by eq. (3.61), together with the interpolation of the face-centered density in the mass flux of the discretized convective term

from eq. (2.15), introduces an inconsistency described in detail and addressed below.

5.1 Numerical consistency of momentum convection term

As outlined in the introduction section 4, many authors have addressed numerical instabilities in various discretizations and two-phase flow methods arising from high-density ratios. In this sub-section, a detailed analysis of the inconsistencies that lead to numerical instabilities by studying the relationships between mass conservation, phase indicator function conservation, and momentum convection is provided. It turns out that the conservative formulation of conservation equations permits us to precisely define equalities that must hold in the mathematical model and its discretization to achieve consistency in the equation system and prevent numerical instabilities.

Bussmann, Kothe, and Sicilian [58] were the first to consider the problem of numerical consistency of the two-phase momentum convective term in the setting of the collocated unstructured Finite Volume method. An expansion on their work by improving the accuracy of the face-centered density evaluation and employing a solution algorithm that allows for an implicit discretization of the convective term, thus removing the CFL condition, is given below. The two-phase momentum convection term from eq. (2.15) using the collocated unstructured finite volume method is discretized as

$$\int_{\Omega_c} \nabla \cdot (\rho \mathbf{v} \mathbf{v}) dV = \int_{\partial\Omega_c} (\rho \mathbf{v} \mathbf{v}) \cdot \mathbf{n} ds = \sum_{f \in F_c} \rho_f F_f \mathbf{v}_f + e_{\rho \mathbf{v}, con}(h^2). \quad (5.1)$$

The second-order discretization error is denoted as $e_{\rho \mathbf{v}, con}(h^2)$, with the $\rho \mathbf{v}$ subscript indicating the equation ($\rho \mathbf{v}$ for momentum eq. (2.15)), *con* subscript the convective term of the equation, and h the discretization length. The convective term in eq. (5.1) has been linearized with respect to the solution variable \mathbf{v} in order to obtain a linear equation system. Here, ρ_f represents the face-center density, and F_f represents the linearized volumetric flux. Details on the flux linearization and temporal integration are given in section 5.2, here the spatial discretization is focused at first.

The discretization 5.1 requires a mass flux $\rho_f F_f$. The volume fraction conservation and the conservation of mass are equivalent if both phases are incompressible. To show this, the

mass conservation equation is written in conservative form in a fixed (time-independent) control volume $\Omega_c \neq \Omega_c(t)$ as

$$\frac{\partial}{\partial t} \int_{\Omega_c} \rho dV = - \int_{\partial\Omega_c} \rho \mathbf{v} \cdot \mathbf{n} dS. \quad (5.2)$$

Applying eq. (2.10) and eq. (2.2) to eq. (5.2) leads to

$$(\rho^- - \rho^+) \frac{\partial}{\partial t} \int_{\Omega_c} \chi dV = -(\rho^- - \rho^+) \int_{\partial\Omega_c} \chi \mathbf{v} \cdot \mathbf{n} dS - \rho^+ \int_{\partial\Omega_c} \mathbf{v} \cdot \mathbf{n} dS, \quad (5.3)$$

with $\int_{\partial\Omega_c} \mathbf{v} \cdot \mathbf{n} dS = \int_{\Omega_c} \nabla \cdot \mathbf{v} dV = 0$ because of eq. (2.2). Dividing eq. (5.3) by $|\Omega_c|$, and using the volume fraction definition eq. (3.60) leads to

$$(\rho^- - \rho^+) \frac{\partial}{\partial t} \alpha_c(t) = -(\rho^- - \rho^+) \frac{1}{|\Omega_c|} \int_{\partial\Omega_c} \chi \mathbf{v} \cdot \mathbf{n} dS. \quad (5.4)$$

The eq. (5.2) implies

$$\frac{\partial}{\partial t} \rho_c(t) = - \frac{1}{|\Omega_c|} \int_{\partial\Omega_c} \rho \mathbf{v} \cdot \mathbf{n} dS. \quad (5.5)$$

An important equality arises from eqs. (5.2), (5.4) and (5.5), namely

$$\partial_t \rho_c(t) = - \frac{1}{|\Omega_c|} \int_{\Omega_c} \rho \mathbf{v} \cdot \mathbf{n} dV = (\rho^- - \rho^+) \partial_t \alpha_c(t) = -(\rho^- - \rho^+) \frac{1}{|\Omega_c|} \int_{\partial\Omega_c} \chi \mathbf{v} \cdot \mathbf{n} dS. \quad (5.6)$$

Selecting $\partial_t \rho_c(t) = (\rho^- - \rho^+) \partial_t \alpha_c(t)$ from eq. (5.6), and integrating over the time interval $[t^n, t^{n+1}]$ leads to

$$\rho_c^{n+1} = (\rho^- - \rho^+) \alpha_c^{n+1} + \rho_c^n - (\rho^- - \rho^+) \alpha_c^n, \quad (5.7)$$

and applying eq. (3.61) at t^n to eq. (5.7) leads to

$$\rho_c^{n+1} = (\rho^- - \rho^+) \alpha_c^{n+1} + \rho^+, \quad (5.8)$$

which is eq. (3.61) at t^{n+1} . Note that the time integration is exact because of the fundamental theorem of calculus. The equality $\partial_t \rho_c(t) = (\rho^- - \rho^+) \partial_t \alpha_c(t)$ from eq. (5.6) can lead to a false conclusion of consistency of the two-phase momentum convection. If other equalities from eq. (5.6) are not upheld, e.g. when the method that advects the phase indicator α_c does not rely on phase-specific fluxes (cf. [34] for a recent review), inconsistencies arise.

Since unstructured finite volumes are bounded by faces S_f ,

$$\frac{1}{|\Omega_c|} \int_{\Omega_c} \rho \mathbf{v} \cdot \mathbf{n} dV = (\rho^- - \rho^+) \frac{1}{|\Omega_c|} \int_{\partial\Omega_c} \chi \mathbf{v} \cdot \mathbf{n} dS$$

from eq. (5.6) is rewritten as

$$\sum_{f \in F_c} \int_{S_f} \rho \mathbf{v} \cdot \mathbf{n} dS = (\rho^- - \rho^+) \sum_{f \in F_c} \int_{S_f} \chi \mathbf{v} \cdot \mathbf{n} dS, \quad (5.9)$$

and discretize it further using second-order-accurate face-averages, resulting in

$$\begin{aligned} \sum_{f \in F_c} \rho_f F_f + O_{\rho,con}(h^2) &= \sum_{f \in F_c} (\rho^- - \rho^+) \frac{F_f}{|\mathbf{S}_f|} \int_{S_f} \chi dS + O_{\alpha,con}(h^2) \\ &= \sum_{f \in F_c} (\rho^- - \rho^+) F_f \alpha_f + O_{\alpha,con}(h^2), \end{aligned} \quad (5.10)$$

with $\phi_f := \frac{1}{|S_f|} \int_{S_f} \phi dS$ defining the face-average associated to the centroid of each face S_f . In eq. (5.10), $F_f := \mathbf{v}_f \cdot \mathbf{S}_f$ is the linearized volumetric flux given by the velocity \mathbf{v} from the discretized single-field Navier-Stokes equations eqs. (2.2) and (2.15). Equation (5.10) reveals an important fact: the mass flux $\rho_f F_f$ - necessary for the discretization of the two-phase momentum convective term eq. (5.1) - must be linearly proportional to the phase-specific volumetric flux $F_f \alpha_f$ used to advect the phase indicator α_c , with $(\rho^- - \rho^+)$ as the proportionality coefficient. This consistency is not ensured by any two-phase flow simulation method that does not solve an advection equation for the volume fractions using a flux-based discretization method.

The unstructured LENT method [2, 35, 85] is extended to ensure that the condition from eq. (5.10) is upheld. Before describing the numerical method in detail, a discussion on a verification case of a droplet advected in a constant velocity field is given.

5.1.1 Verification case: droplet translating with constant velocity

The Euler explicit collocated unstructured FV discretization of eq. (5.2) is

$$\rho_c^{n+1} = \rho_c^n - \frac{\Delta t}{|\Omega_c|} \sum_{f \in F_c} \rho_f^n F_f^n. \quad (5.11)$$

The two-phase momentum advection is modeled using eq. (2.15) with a prescribed initial constant velocity and without forces on the r.h.s, namely

$$\partial_t(\rho\mathbf{v}) + \nabla \cdot (\rho\mathbf{v}\mathbf{v}) = 0. \quad (5.12)$$

Without forces on the r.h.s. of eq. (5.12), the initial constant velocity should remain spatially constant. Therefore, a numerically consistent unstructured collocated FV discretization of the two-phase momentum convection equation (eq. (5.12)) must ensure that no artificial acceleration or deceleration occurs. For example, just like eq. (5.11), the Euler explicit discretization of eq. (5.12) is

$$\rho_c^{n+1}\mathbf{v}_c^{n+1} = \rho_c^n\mathbf{v}_c^n - \frac{\Delta t}{|\Omega_c|} \sum_{f \in F_c} \rho_f^n F_f^n \mathbf{v}_f^n. \quad (5.13)$$

Given a consistent discretization, the velocity field remains spatially constant, so

$$\mathbf{v}_f^n = \mathbf{v}_c^n, \quad (5.14)$$

which is, of course, ensured for the initial spatially constant velocity ($\mathbf{v}_f^0 = \mathbf{v}_c^0$). Equation (5.14), applied to eq. (5.13), results in

$$\rho_c^{n+1}\mathbf{v}_c^{n+1} = \mathbf{v}_c^n \left(\rho_c^n - \frac{\Delta t}{|\Omega_c|} \sum_{f \in F_c} \rho_f^n F_f^n \right), \quad (5.15)$$

and dividing by ρ_c^{n+1} finally gives

$$\mathbf{v}_c^{n+1} = \frac{\mathbf{v}_c^n \left(\rho_c^n - \frac{\Delta t}{|\Omega_c|} \sum_{f \in F_c} \rho_f^n F_f^n \right)}{\rho_c^{n+1}}. \quad (5.16)$$

As there are no forces on the r.h.s. of eq. (5.12), the velocity should not be changed simply by advecting the two-phase momentum, i.e.

$$\mathbf{v}_c^{n+1} = \mathbf{v}_c^n, \quad (5.17)$$

and this condition is ensured in eq. (5.16) if

$$\frac{\rho_c^n - \frac{\Delta t}{|\Omega_c|} \sum_{f \in F_c} \rho_f^n F_f^n}{\rho_c^{n+1}} = 1, \quad (5.18)$$

which is equivalent to eq. (5.11): the Euler explicit discretization of the mass conservation equation. Consequently, a numerically consistent discretization of the momentum convection equation in this verification case requires the new cell-centered density ρ_c^{n+1} to be computed by solving a mass conservation equation.

Modern unstructured geometric flux-based VOF ([6, 69, 86, 87], see [34] for a recent review) potentially ensure this property, since they solve the conservative formulation of the volume fraction advection equation [34] for α_c^{n+1} by computing phase-specific fluxed volumes, scale the phase-specific fluxed volumes to compute the mass flux, and use the cell-centered volume fraction α_c^{n+1} to compute ρ_c^{n+1} with eq. (3.61). However, the temporal discretization scheme used in the momentum equation for the convective term must be consistent with the integration of the fluxed phase-specific volumes, used to obtain α_c^{n+1} . Even if the mass flux can be computed by scaling the phase-specific fluxed volumes with δt , any difference between the temporal integration schemes used for the volume fraction and momentum equations, or any flux limiting in the momentum equation, cause inconsistencies. Additionally, the $\alpha_c^{n+1} \in [0, 1]$ must hold near machine epsilon. Any correction to α_c^{n+1} performed after the numerical solution of the volume fraction advection equation, that bounds α_c^{n+1} within $[0, 1]$, results in a discrepancy between ρ_c^{n+1} computed using the mass flux that gives unbounded α_c^{n+1} , and the ρ_c^{n+1} computed from the a-posteriori bounded α_c^{n+1} using eq. (3.61).

It is important to note that if the pressure gradient is included on the r.h.s of eq. (5.12), any error in \mathbf{v}_c^{n+1} will result in non-zero source terms on the r.h.s. of the resulting pressure equation, in the $p - \mathbf{v}$ coupling algorithm. Since the pressure gradient enforces $\nabla \cdot \mathbf{v} = 0$ ($\sum_{f \in F_c} F_f = 0$ on the discrete level), this results in artificial velocities similar to parasitic currents caused by the surface tension force.

Bussmann, Kothe, and Sicilian [58] have utilized the consistency of the Volume-of-Fluid method and the availability of phase-specific volumetric fluxes in the VOF method to first solve eq. (5.12) explicitly in the first step, followed by the second step that includes volume and surface forces. The approach from Bussmann, Kothe, and Sicilian [58] cannot be applied without modifications to the Level Set method, the Front Tracking method, their hybrids, or any other collocated FV two-phase flow simulation method that does not rely on phase-specific volumetric fluxes to discretely advect volume fractions. If the phase-specific volumetric fluxes (or volumes) are not calculated by the method, they cannot be used to construct a

consistent mass flux. The solution algorithm for high density ratios that proposed in this study avoids the CFL condition imposed by Bussmann, Kothe, and Sicilian [58] and increases the accuracy of the face-centered density ρ_f required by the mass flux, and it is applicable to any multiphase flow simulation method that utilizes the single-field formulation of the Navier-Stokes equations.

5.2 A semi-implicit solution algorithm for high-density ratios

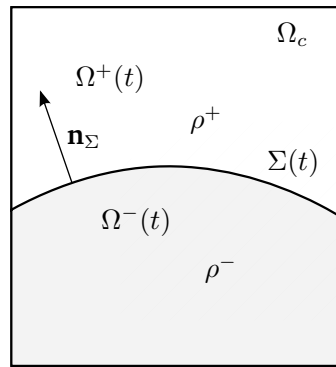


Figure 9: A two-phase fixed control volume Ω_c separated by the interface $\Sigma(t)$.

Section 5.1 provides the formal reasoning behind solving the mass conservation equation (or its equivalent) for ρ_c^{n+1} . Since Ghods and Herrmann [51] introduced an "auxiliary" mass conservation equation, other researchers have adopted this approach, with the main difference in the way the face-centered (mass flux) density ρ_f is evaluated both in the discretized mass conservation equation (eq. (5.2)) and the discretized momentum equation (eq. (2.15)).

The condition given by eq. (5.18), derived from eqs. (5.16) and (5.17) can be fulfilled only if the same face-centered (mass flux) density is used when discretizing the auxiliary mass conservation and momentum equations. Going one step further, the volumetric flux F_f must also be the same in the discretized auxiliary mass conservation and momentum equations. Put together, the mass flux in the auxiliary discretized mass conservation equation must be equal to the mass flux in the discretized momentum conservation equation: this is

the requirement for the mass flux consistency, mentioned throughout the literature.

It is relevant to point out that the same model for the single-field density given by eq. (3.61) is used throughout the literature. The basis of this model is mass conservation, and this fundamental principle further leads to an interesting conclusion regarding the evaluation of the face-centered (mass flux) density ρ_f in the discretized mass and momentum conservation equations. The face centered density is evaluated differently throughout scientific publications reviewed in section 4, and here we show that there is a strict relationship between the phase indicator and the face centered density ρ_f .

Consider the fixed control volume Ω_c in fig. 9, that is separated by the fluid interface $\Sigma(t)$ into two parts, occupied by fluids $\Omega^\mp(t)$. The single-field density model given by eq. (2.10) is adopted in every publication reviewed in section 4, and in the rest of the scientific literature on two-phase flow simulations. The mass conservation principle together with the single-field density model (eq. (2.10)) give

$$\frac{d}{dt} \int_{\Omega_c} \rho dV = - \int_{\partial\Omega_c} \rho \mathbf{v} \cdot \mathbf{n} dS = - \int_{\partial\Omega_c} [\rho^- \chi + \rho^+ (1 - \chi)] \mathbf{v} \cdot \mathbf{n} dS. \quad (5.19)$$

The equality of surface integrals in eq. (5.19),

$$\int_{\partial\Omega_c} \rho \mathbf{v} \cdot \mathbf{n} dS = \int_{\partial\Omega_c} [\rho^- \chi + \rho^+ (1 - \chi)] \mathbf{v} \cdot \mathbf{n} dS,$$

demonstrates that the mass flux of the single-field density over $\partial\Omega_c$ is determined by the constant densities ρ^\mp and the phase indicator given by eq. (2.9), if eq. (2.10) is used to model the single-field density. In other words, the single-field density at $\partial\Omega_c$ should be computed using the phase indicator as done on the r.h.s. of eq. (5.19), otherwise the mass conservation of the single-field density model given by eq. (2.10) will not be upheld. This relevant condition transfers to the discrete level, leading to an interesting consequence for the computation of the face-centered (mass flux) density, that has so far been computed in many ways throughout the literature.

Specifically, when the surface integrals in eq. (5.19) are discretized using the unstructured

collocated finite volume method,

$$\begin{aligned}
\sum_{f \in F_c} \rho_f F_f &= \sum_{f \in F_c} \left[\rho^- \left(\int_{S_f} \chi dS \right) \mathbf{v}_f \cdot \hat{\mathbf{S}}_f + \rho^+ \left(\int_{S_f} dS \right) \mathbf{v}_f \cdot \hat{\mathbf{S}}_f - \rho^+ \left(\int_{S_f} \chi dS \right) \mathbf{v}_f \cdot \hat{\mathbf{S}}_f \right] \\
&= \sum_{f \in F_c} \left[\rho^- \frac{\|\mathbf{S}_f\|}{\|S_f\|} \left(\int_{S_f} \chi dS \right) \mathbf{v}_f \cdot \hat{\mathbf{S}}_f + \rho^+ \left(\int_{S_f} dS \right) \mathbf{v}_f \cdot \hat{\mathbf{S}}_f \right. \\
&\quad \left. - \rho^+ \frac{\|\mathbf{S}_f\|}{\|S_f\|} \left(\int_{S_f} \chi dS \right) \mathbf{v}_f \cdot \hat{\mathbf{S}}_f \right] \\
&= \sum_{f \in F_c} [\rho^- \alpha_f + \rho^+ (1 - \alpha_f)] F_f,
\end{aligned} \tag{5.20}$$

where

$$\alpha_f := \frac{1}{|S_f|} \int_{S_f} \chi dS \equiv \frac{|\Omega^-(t) \cap S_f|}{|S_f|} \tag{5.21}$$

is the *area fraction* of the face $S_f \subset \partial\Omega_c$, i.e. the ratio of the area of S_f submerged in $\Omega^-(t)$, and the total face-area $|S_f|$. Further, $\|\mathbf{S}_f\| \equiv |S_f|$, and F_f is the volumetric flux in eq. (5.20).

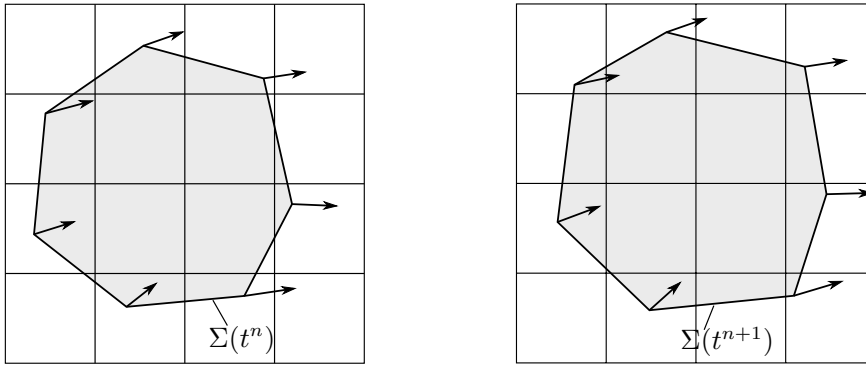
An important consequence of eq. (5.20) is the requirement for the evaluation of the face-centered (mass flux) density, necessary for ensuring the numerical consistency of the single-field two-phase momentum convection. Equation (5.20) requires all methods³ that define ρ using eq. (3.61) to either compute ρ_f using the area fractions or $\int_{S_f} \chi dS$ from eq. (5.20), or to achieve this equivalently when computing ρ_c^{n+1} from the advected volume fractions α_c^{n+1} , which is possible for the flux-based VOF methods [34].

Another important realization is that eq. (5.20) is valid at any time t - which is very relevant for the semi-implicit discretization developed within the ρ LENT method, that applies eq. (5.20) at t^{n+1} .

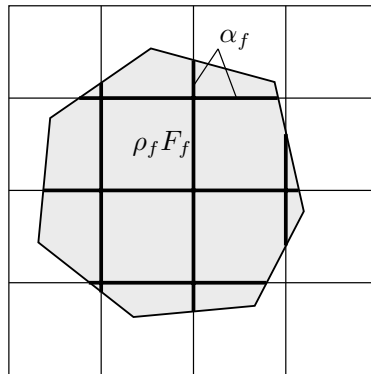
Any simulation method that relies on the collocated unstructured FV discretization of single-field two-phase Navier-Stokes equations, but does not advect the phase indicator by solving an advection equation using phase-specific volumetric fluxes, does not provide the phase-specific volumetric fluxes for the approximation of the mass fluxes needed to ensure the consistency of the two-phase momentum transport. This, however, does not infer that

³All two-phase flow simulation methods encountered use eq. (3.61).

eq. (5.20) cannot be applied. The idea of using an auxiliary mass conservation equation introduced by [51], made into a formal requirement by eqs. (5.16) and (5.17), allows the use of eq. (5.20): α_f can be computed regardless of the approximation of the fluid interface $\Sigma(t)$ and the method used to advect it.



(a) Interface Σ at t^n and t^{n+1} and the respective $\Omega^-(t^n)$ and $\Omega^-(t^{n+1})$ in gray color, used to compute α_c^n and α_c^{n+1} , that are further used to compute ρ_c^n and ρ_c^{n+1} in an inconsistent way.



(b) Interface at $\Sigma(t^{n+1})$ used to compute α_f^{n+1} , then ρ_f^{n+1} and finally ρ_c^{n+1} in a consistent way, by solving a mass conservation equation.

Figure 10: Updating the face-centered (mass flux) density in the ρ LENT method.

Similar to other contemporary methods, the ρ LENT method also first advects the interface

using the velocity from the previous time step as shown in the left image of fig. 10a, resulting in the new position of the interface shown in the right image in fig. 10a, that is then used to geometrically calculate the face-centered density ρ_f^{n+1} , by calculating area fractions α_f^{n+1} from the interface approximation, as shown in fig. 10b. The face-centered density ρ_f^{n+1} and the volumetric flux F_f^o are then used to update the cell-centered density ρ_c^{n+1} by solving a mass conservation equation. The index o in the volumetric flux refers to the linearization of the convective term in the momentum equation. The same mass flux $\rho_f^{n+1} F_f^o$ is used in the implicitly discretized momentum conservation equation. The pressure-velocity coupling algorithm iterates the linearized volumetric flux F_f^o to F_f^{n+1} . Finally, the cell-centered velocity \mathbf{v}_c^{n+1} is obtained, which is used to evolve the fluid interface in the next time step, from t^{n+1} to t^{n+2} . At this point, the numerically consistent cell-centered density ρ_c^{n+1} has served its purpose and is reset according to eq. (3.61), using α_c approximated from signed distances [2], to make it consistent again with the fluid interface approximation.

In the original Front-Tracking method, the density is updated utilizing the new position of marker points (the approximated interface) [88]. After the velocity field in the current step is computed, the position of marker points in the new time step can be updated immediately by

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^n, \quad (5.22)$$

where $\mathbf{x}_p, \mathbf{v}_p$ indicate the position and interpolated velocity of marker points respectively, and Δt is the time step length. The advection of marker points along Lagrangian trajectories eventually corrupts the triangular mesh, leading to discrepancies in the ratios of triangular angles and areas and self-intersections of the triangular mesh. The original Front Tracking method [37] deals with this by redistributing marker points based on quality criteria imposed on the triangular mesh, which involves manipulating the connectivity of the triangular mesh.

Contrary to original Front Tracking [37], the LENT method reuses the principles from Level Contour Reconstruction Method (LCRM) / Local Front Reconstruction Method (LFRM) methods [38, 46, 83, 84] and reconstructs the interface using an iso-surface reconstruction algorithm. The iso-surface reconstruction does not add/delete marker points locally by changing the connectivity of the triangular surface mesh; it reconstructs the entire interface in the solution domain as an iso-surface. Following the strategy from LCRM / LFRM, the

physics of the problem determines the iso-surface reconstruction frequency. The LENT method uses the marching tetrahedra [89] algorithm to enable the iso-surface reconstruction on unstructured meshes.

Once the marker points are advected and redistributed, the cell density is updated depending on \mathbf{x}_p^{n+1} , namely

$$\rho^{n+1} = \rho(\mathbf{x}_p^{n+1}). \quad (5.23)$$

The face-centered density used for the mass flux is then interpolated by the LENT method from densities of two adjacent cells. Contrary to LENT, the face-centered density is updated by ρ LENT using the phase indicator approximated at each cell-face by an area fraction. A 2D interface is depicted in fig. 10b, where α_f^{n+1} is the area fraction at t^{n+1} : the ratio of the cell-face area submerged in the phase $\tilde{\Omega}^-(t^{n+1}) \approx \Omega^-(t^{n+1})$, and the total face area $|S_f|$. More precisely, the area fraction α_f^{n+1} is computed by the ρ LENT method using a second-order accurate approximation from signed distances [90], used in [2] to approximate the volume fraction α_c (see eq. (3.60)). The Level Set component of the LENT method [35] calculates signed distances from the triangular surface mesh that approximates the interface $\tilde{\Sigma}(t^{n+1}) \approx \Sigma(t^{n+1}) := \partial\tilde{\Omega}^-(t^{n+1})$. With the narrow band approach from [35], the signed distances can be computed efficiently at any point in a close vicinity of $\tilde{\Sigma}(t)$. The original LENT method [35] computes signed distances at cell-centers and cell corner-points, and the proposed ρ LENT additionally computes signed distances at face centers. Each face S_f is triangulated using its centroid \mathbf{x}_f , as shown in fig. 11. The face centroid \mathbf{x}_f , together with the two successive cell-corner points that belong to the face S_f , $\mathbf{x}_{f,i}$, $\mathbf{x}_{f,i+1}$, forms a triangle $(\mathbf{x}_f, \mathbf{x}_{f,i}, \mathbf{x}_{f,i+1})$. Face-triangles may be partially submerged in the phase $\tilde{\Omega}^-(t^{n+1})$, in which case the submerged area of the triangle is computed using the nearest signed distances to $\tilde{\Sigma}(t^{n+1})$ from the triangle points $(\mathbf{x}_f, \mathbf{x}_{f,i}, \mathbf{x}_{f,i+1})$, namely $(\psi_f, \psi_{f,i}, \psi_{f,i+1})$, as shown in fig. 11. The second-order approximation developed in [90] is used here for computing the area fraction of a triangle submerged in $\tilde{\Omega}^-(t^{n+1})$. Any other second-order method can be applied. For example, a linear interpolation of signed distances along the edges of the triangle may be used equivalently, or a geometrical intersection between $\tilde{\Omega}^-(t^{n+1})$ and the triangle. The total

submerged area of the face S_f is then the sum of the submerged areas of face-triangles

$$A_f^{n+1} := |\Omega^-(t^{n+1}) \cap S_f| = \sum_{k \in T_f} |\Omega^-(t^{n+1}) \cap T_k|, \quad (5.24)$$

where T_f is the set of indexes of the triangles in the triangulation of the face S_f . As mentioned above, any other two-phase flow simulation method that discretizes single-field Navier-Stokes equations but does not utilise phase-specific fluxes can be adapted to compute $|\Omega^-(t^{n+1}) \cap T_k|$.

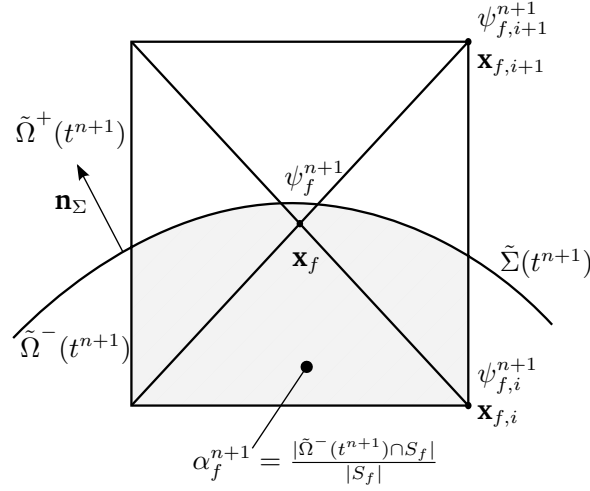


Figure 11: Computing area fractions from signed distances in the method.

The area fraction α_f^{n+1} is then computed as

$$\alpha_f^{n+1} := \frac{|\tilde{\Omega}^-(t^{n+1}) \cap S_f|}{|S_f|} = \frac{A_f}{|S_f|}, \quad (5.25)$$

as shown in fig. 11. Once the area fraction α_f^{n+1} is approximated, it is used to compute the face-centered densities required by eq. (5.20), namely

$$\rho_f^{n+1} = \alpha_f^{n+1} \rho^- + (1 - \alpha_f^{n+1}) \rho^+, \quad (5.26)$$

at the new time step, because the interface has been advected forward in time to t^{n+1} with the available velocity \mathbf{v}^n . The discretized continuity equation (eq. (5.11)) then attains the

form

$$\rho_c^{o+1} = \rho_c^n + \frac{\Delta t}{|V_{\Omega_c}|} \sum_f \rho_f^{n+1} F_f^o, \quad F_f^o = \mathbf{v}_f^o \cdot \mathbf{S}_f. \quad (5.27)$$

It is important to note that, although ρ_f^{n+1} appears in eq. (5.27), ρ LENT does not use an implicit discretization for eq. (5.27): ρ_f^{n+1} is geometrically computed from the fluid interface approximation $\tilde{\Sigma}^{n+1}$, so eq. (5.27) is solved exactly. The exact (iterative) evaluation of cell-center density at loop $o + 1$, i.e. ρ_c^{o+1} from eq. (5.27), alongside eq. (5.16), further infers the possibility of *exact numerical consistency* for the discretized convective term in the single-field momentum equation, which is in fact achieved and supported by the results.

In addition to density, the viscosity is updated utilizing the area fraction α_f . Note that there is no need to calculate the cell-centered viscosity for the unstructured FV discretization, only the face-centered viscosity is updated as follows

$$\mu_f^{n+1} = \alpha_f^{n+1} \rho^- \nu^- + (1 - \alpha_f^{n+1}) \rho^+ \nu^+. \quad (5.28)$$

The non-linearity of the convective term in the momentum equation eq. (2.15), namely $\rho \mathbf{v} \mathbf{v}$, is usually linearized when solving the single-field Navier-Stokes equations using the unstructured Finite Volume method. The convective term is discretized as

$$\int_{\Omega_c} \nabla \cdot (\rho \mathbf{v} \mathbf{v}) dV \approx \sum_{f \in F_c} \rho_f^{n+1} F_f^o \mathbf{v}_f^{n+1}. \quad (5.29)$$

Numerical consistency imposed by eq. (5.6) does not depend on the implicit or explicit discretization: the proportionality between the mass flux and the phase-specific flux, and the equivalence of the mass flux in the mass conservation equation and the momentum transport equation must both hold at any time, and in any iteration of the solution algorithm. Therefore, the requirement given by eqs. (5.16) and (5.17), is valid for an implicit discretization as well.

The volumetric flux F_f^o is initialized to F_f^n and iterated within the SAAMPLE [2] pressure-velocity coupling algorithm loop until $o = n + 1$ is reached. The ρ LENT algorithm is outlined in algorithm 5 and it extends the SAAMPLE algorithm [2]. It is relevant to note that F_f^o is iterated from F_f^n to F_f^{n+1} and p^o is solved for from p^n to p^{n+1} such that the discrete incompressibility condition $\sum_{f \in F_c} F_f^{n+1}$ is ensured.

Algorithm 5 The ρ LENT solution algorithm.

- 1: **while** simulation time \leq end time **do**
 - 2: Advect the interface to $\tilde{\Sigma}^{n+1}$. ▷ [35]
 - 3: Compute the signed-distance field ψ^{n+1} from $\tilde{\Sigma}^{n+1}$ at $\mathbf{x}_c, \mathbf{x}_f, \mathbf{x}_p$ in the narrow-band. ▷ [35]
 - 4: Compute α_c^{n+1} from $\psi_c^{n+1}, \psi_p^{n+1}$. ▷ [2]
 - 5: Compute the area fraction α_f^{n+1} from the signed distance fields $\psi_f^{n+1}, \psi_p^{n+1}$. ▷ Figure 11
 - 6: Compute the face-centered densities ρ_f^{n+1} using α_f^{n+1} . ▷ Equation (5.26)
 - 7: **while** F_f^o does not converge or $o < o_{max}$ **do**
 - 8: Solve the continuity equation using $\rho_f^{n+1} F_f^o$ for cell-centered densities ρ_c^{o+1} . ▷ Equation (5.27)
 - 9: **while** $r > tol_s$ and $i < i_{max}$ **do**
 - 10: Use ρ_c^{o+1} and $\rho_f^{n+1} F_f^o$ in $p - \mathbf{v}$ coupling to compute $\mathbf{v}_c^{i+1}, F_f^{i+1}$. ▷ [2] and eq. (5.29)
 - 11: **end while**
 - 12: **end while**
 - 13: Make ρ_c^{n+1} consistent with $\tilde{\Sigma}^{n+1}$, i.e. $\rho_c^{n+1} = \alpha_c^{n+1} \rho^- + (1 - \alpha_c^{n+1}) \rho^+$.
 - 14: Make μ_c^{n+1} consistent with $\tilde{\Sigma}^{n+1}$, i.e. $\mu_c^{n+1} := \alpha_c^{n+1} \rho^- \nu^- + (1 - \alpha_c^{n+1}) \rho^+ \nu^+$.
 - 15: **end while**
-

The $p - \mathbf{v}$ coupling - mentioned in the step 8 in algorithm 5 - requires some further explanation. The semi-implicit discretization (with the convective term linearized as an explicit mass-flux and implicit velocity) of the single-field momentum equation using the implicit collocated unstructured finite volume method [17, 19], results in

$$a_c \mathbf{v}_c^{n+1} + \sum_{k \in N_c} a_k \mathbf{v}_k^{n+1} = -(\nabla p)_c^{n+1} - [(\nabla \rho)^{n+1} \cdot (\mathbf{g} \cdot \mathbf{x})]_c + (\mathbf{f}_\Sigma)_c^{n+1}, \quad (5.30)$$

where N_c is the index-set of cells that are face-adjacent to cell Ω_c , and the total pressure is expressed using the dynamic and the hydrostatic pressure. The diagonal coefficient a_c corresponds to the cell Ω_c , and k denotes the coefficients contributed from cells that are face-adjacent to Ω_c . We discretize the surface tension force $(\mathbf{f}_\Sigma)_c^{n+1}$ using the semi-implicit model from [2].

Equation (5.30) is also discretized semi-implicitly, because of the linearized convective term, that contributes the volumetric flux F_f to the $a_{c,k}$ coefficients in eq. (5.30). Linearizing the convective term introduces a need for iteration. Iterations are also introduced by splitting eq. (5.30) into two equations: one for \mathbf{v}_c^{n+1} , and another for \mathbf{p}_c^{n+1} . Dividing the equation

eq. (5.30) with a_c and applying the discrete divergence $\nabla_c \cdot$, results in the pressure equation

$$\begin{aligned} \sum_{f \in F_c} \left(\frac{1}{a_c} \right)^o (\nabla p)_f^{i+1} \cdot \mathbf{S}_f &= \sum_{f \in F_c} \left(\frac{1}{a_c} \right)^o [\mathbf{H}(\mathbf{v}^i)]_f \cdot \mathbf{S}_f + \sum_{f \in F_c} \left(\frac{1}{a_c} \right)^o [(\nabla \rho)^i \cdot (\mathbf{g} \cdot \mathbf{x})]_f \cdot \mathbf{S}_f + \\ &\quad \sum_{f \in F_c} \left(\frac{1}{a_c} \right)^o \sigma \kappa_f^{n+1} (\nabla \alpha)_f^i \cdot \mathbf{S}_f, \end{aligned} \quad (5.31)$$

where we use the CSF model [27] to model the surface tension force $(\mathbf{f}_\Sigma)_f \approx \sigma \kappa_f (\nabla \alpha)_f$. The discrete divergence-free condition imposed on \mathbf{v}_c^{n+1} in eq. (5.30) results in the divergence-free volumetric flux

$$\sum_{f \in F_c} F_f^o = 0, \quad (5.32)$$

used as the control variable for the convergence of outer iterations o by the SAAMPLE algorithm [2]. The outer iterations o are used for linearizing the volumetric flux as described above and contribute the volumetric flux to the coefficients $a_{c,k}$, from eq. (5.30), while $\mathbf{H}(\mathbf{v})$ in eq. (5.31) is the contribution of convection and diffusion operators from face-adjacent cells in eq. (5.30). Note that $(\nabla \rho)_{c,f}^{n+1}$ and the implicit part of $(\mathbf{f}_\sigma^{n+1})_{f,c}$ are known at t^{n+1} from $\mathbf{f}_\sigma^{n+1} := \mathbf{f}_\sigma^{n+1}(\{\mathbf{x}_p^{n+1}\}_{p \in P})$, and eq. (5.27).

This segregated solution for $(p_c^{n+1}, \mathbf{v}_c^{n+1})$ is standard in the context of collocated unstructured finite volume method [17]: the inner iterations and the assembly of the pressure equation originates from the PISO algorithm [31], the outer iterations originate from the SIMPLE algorithm [30], and the tolerance-based control of outer iterations is described in detail in [2]. In addition, the implementations of the LENT method [35], the SAAMPLE algorithm [2] and the ρ LENT method are publicly available [56]. This description, the details on the tolerance-based outer iteration control in [2], and the publicly available implementation in OpenFOAM[®], provide sufficient information for an interested reader willing to understand or further extend the methodology.

5.3 Volume correction method

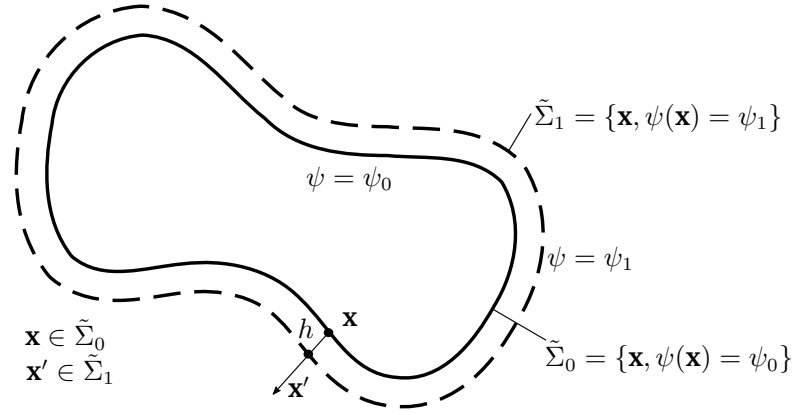


Figure 13: The volume correction method: iso-value compensates the volume-change.

Level Set / Front Tracking methods are Lagrangian / Eulerian methods that kinematically evolve the fluid interface without utilising fluxes through control-volume boundaries and are therefore inherently not mass(volume)-conservative. Rising bubble and oscillating droplet simulations are presented in the results section to demonstrate the benefits of the ρ LENT method for stronger momentum interaction between fluid phases that have strongly different densities. Mass conservation is crucial for accurately simulating rising bubbles (Hua and Lou [4], Singh and Shyy [91], Hua, Stene, and Lin [92], and Pivello et al. [93]). In particular, Hua and Lou [4] conducted comparative analyses which revealed that mass conservation carries equivalent importance to mesh resolution and domain size in influencing the accuracy. The Front reconstruction in the LENT method [2, 35] uses marching tetrahedrons with linear interpolation of the iso-surface root-points Treece, Prager, and Gee [89], that causes volume loss. To demonstrate the benefits of the proposed ρ LENT method for handling high density ratios with stronger interface deformation and momentum exchange, we ensure volume conservation using by extending/contracting the Front with a modified iso-value.

Figure fig. 13 depicts the volume correction at time step n , where $\tilde{\Sigma}_0$ denotes the Front at the time step n . The value $\psi_0 = 0$ is the iso-value used to reconstruct the $\tilde{\Sigma}_0$ at t^n . From $\tilde{\Sigma}_0$ that contains volume-conservation errors, we compute the corrected Front $\tilde{\Sigma}_1$, as $\tilde{\Sigma}_0$ extended

in the normal direction by h . We consider volume loss, with no loss of generality in the case of volume gain. For sufficiently small h , for any $\mathbf{x} \in \tilde{\Sigma}_0$, we define $\mathbf{x}' := \mathbf{x} + h\mathbf{n}_\Sigma(\mathbf{x})$, $\mathbf{x}' \in \tilde{\Sigma}_1$. The linear Taylor-series approximation

$$\psi(\mathbf{x}') \doteq \psi(\mathbf{x}) + \nabla\psi(\mathbf{x}) \cdot h\mathbf{n}_\Sigma(\mathbf{x}), \quad (5.33)$$

with $\psi(\mathbf{x}) = 0$ by the definition of an iso-surface $\forall \mathbf{x} \in \tilde{\Sigma}_0$, results in

$$\psi(\mathbf{x}') \doteq h\nabla\psi(\mathbf{x}) \cdot \mathbf{n}_\Sigma(\mathbf{x}). \quad (5.34)$$

Since Level Set / Front Tracking ensures $\nabla\psi(\mathbf{x}) = \mathbf{n}_\Sigma$ and thus $\nabla\psi(\mathbf{x}) \cdot \mathbf{n}_\Sigma(\mathbf{x}) = 1$ by geometrically re-distancing ψ from the reconstructed Front $\tilde{\Sigma}_0^n$,

$$\psi(\mathbf{x}') = h. \quad (5.35)$$

The height h is expressed from the change in volume between $\tilde{\Sigma}_0$ and $\tilde{\Sigma}_1$

$$\begin{aligned} \int_{\tilde{\Sigma}_0} h \, dS &= V_{target} - V_{ini}, \\ h &= \frac{V_{target} - V_{ini}}{\int_{\tilde{\Sigma}_0} 1 \, dS}, \end{aligned} \quad (5.36)$$

Volume V_{target} is the target volume and known before the reconstruction, and we aim to recover V_{target} 's corresponding front $\tilde{\Sigma}_1$. The volume V_{ini} is computed from the initially reconstructed $\tilde{\Sigma}_0$. If volume loss really occurred, then $V_{target} > V_{ini}$, and $h > 0$, so we extend $\tilde{\Sigma}_0$ in the direction of \mathbf{n}_Σ by reconstructing an iso-surface $\tilde{\Sigma}_1 = \{\mathbf{x}' : \psi(\mathbf{x}') = h\}$. However, if volume gain occurred, $V_{target} < V_{ini}$, so $h < 0$, and reconstructing an iso-surface $\tilde{\Sigma}_1 = \{\mathbf{x}' : \psi(\mathbf{x}') = h\}$ shrinks $\tilde{\Sigma}_0$ in the normal direction.

The volume V_{ini} is computed geometrically [85] as

$$V_{ini} = \frac{1}{3} \left| \sum_{e=1}^{N_{\tilde{\Sigma}_0}} \mathbf{x}_e \cdot \mathbf{S}_e \right|, \quad (5.37)$$

where $N_{\tilde{\Sigma}_0}$ is the number of triangles in $\tilde{\Sigma}_0$, \mathbf{x}_e is a centroid, and \mathbf{S}_e the area-normal vector of the e -th triangle in $\tilde{\Sigma}_0$. At reconstruction time step t^n , the front is first reconstructed using the iso-value $\psi_0 = 0$, as shown in fig. 13. The volume $V^n(\psi_0 = 0)$ is then calculated

w.r.t eq. (5.37). Since the loss/gain of volume between successive reconstructions is small ($\mathcal{O}(10^{-4})$), the compensated extension/contraction is uniformly distributed across the Front. The iso-value adjustment given by eq. (5.36) is then discretized as

$$h = \frac{V(\tilde{\Sigma}_1) - V_{ini}}{\sum_{e=1}^{N_{\tilde{\Sigma}_0}} |\mathbf{S}_e|}, \quad (5.38)$$

in which $|\mathbf{S}_e|$ denotes the area of the e -th triangle. Reconstruction with the new iso-value $\psi_1 = h^n$ generates a volume-conserved Front, as illustrated by the solid line on the right in fig. 13.

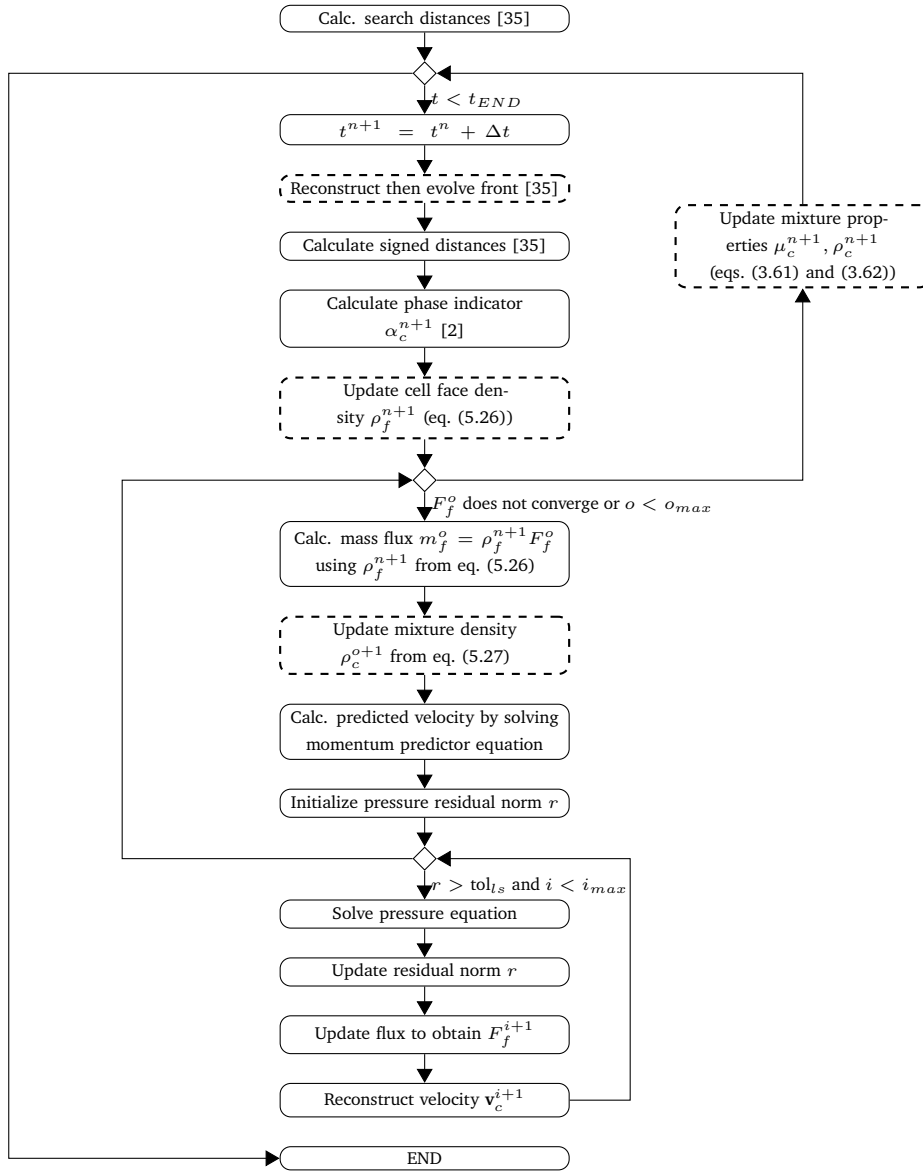


Figure 12: Flowchart of the ρ LENT method. The dashed blocks denote the new and modified elements of the SAAMPLE method [2]. The indices o_{max} and i_{max} in the flowchart indicate the maximal iteration numbers for the outer and inner loop, respectively, while tol_{ls} denotes the prescribed linear solver tolerance.

6 Verification and validation

The hybrid Level Set / Front Tracking method is not strictly volume conservative, and volume errors arise from three sources. First, the iso-surface reconstruction - that handles the topological changes of the fluid interface - introduces volume errors by interpolating the level-set function. This error source can be reduced using higher-order level set function interpolation. Second, the Front Tracking method approximates the fluid interface as a surface triangulation and advects the interface in a co-moving reference frame by displacing the triangulation points along Lagrangian trajectories. The volume errors introduced by Front Tracking are reducible significantly by a second (or higher)-order temporal integration of the Lagrangian displacements. The third source of volume conservation errors is the phase-indicator model: Volume fractions are approximated from signed distances stored at cell centers and cell-corner points in previous work [2]; a more accurate geometrical intersection between the Front and the volume mesh is being investigated in [85]. The volume conservation of the hybrid Level Set / Front Tracking method depends on the physics of the problem. For the verification problems, the ρ LENT method recovers very low maximal relative volume conservation errors of $5.13 \cdot 10^{-4}$ for the coarsest resolution of only 6 cells per droplet diameter and $5.49 \cdot 10^{-5}$ for the finest resolution of 26 cells per droplet diameter. The volume conservation errors of such small magnitude have no effect on the numerical stability of the two-phase momentum convection term, so their detailed visualization is omitted for brevity.

Secondary data presented in this section in the form of diagrams and tables [94], the snapshot of the LENT implementation used in this manuscript [95], and the active development repository of the LENT method as an OpenFOAM[®] module [96] are publicly available.

6.1 Time step size

The time step size limit due to the CFL condition is given by

$$\Delta t \leq \Delta t_{CFL} = \frac{h}{U}, \quad (6.1)$$

where h is cell length and U is a characteristic velocity. In the cases, h is the minimum cell size, while U is equal to magnitude of the ambient flow velocity vector, i.e. $U = |\mathbf{v}_a| = 1$. Another restriction for the time step size arises from the propagation of capillary waves on interfaces between two fluids. This time step constraint is firstly introduced by Brackbill, Kothe, and Zemach [27], and afterwards revised by Denner and Wachem [97]. It has the form

$$\Delta t \leq \Delta t_{cw} = \sqrt{\frac{(\rho_d + \rho_a)h^3}{2\pi\sigma}}, \quad (6.2)$$

in which ρ_d and ρ_a are density of droplet and ambient fluid, respectively, σ is the surface tension coefficient. In the case setup procedure, the method devised by Tolle, Bothe, and Marić [2] is followed, i.e., using a compare function

$$\Delta t = \min(k_{cw}\Delta t_{cw}, k_{CFL}\Delta t_{CFL}) \quad (6.3)$$

where k_{cw} and k_{CFL} are arbitrary scale factors between 0 and 1. In the following, $k_{cw} = 0.5$ and $k_{CFL} = 0.2$ are used.

6.2 Translating droplet

Following the setup of Popinet [98], a sphere of radius $R = 0.2$ translates in a rectangular domain having side lengths $L_x = L_y = 5R, L_z = 6R$. The initial position of the sphere's centroid is $C_x = C_y = 0.5, C_z = 0.4$. One corner of the rectangular domain locates in the origin as shown in fig. 14. The boundary conditions of the rectangular domain are set as follows: $\nabla \mathbf{v} = 0$ and $p = 0$ for the outlet, $\mathbf{v} = \mathbf{v}_a$ and zero gradient $\nabla p = 0$ for the pressure at the mantle and the inlet. The initial conditions for internal field is set to $p(t_0) = 0$ and $\mathbf{v}(t_0) = \mathbf{v}_a$. The end time of simulation is set to $t_{end} = 0.41$ s, which corresponds to a droplet displacement of one diameter.

Two groups of cases are tested to verify the ρ LENT method, their parameters are listed in table 1. For the first group, only the advection of momentum and pressure term are considered, and the ambient flow has a constant density $\rho_a = 1$, while the density of the droplet ρ_d varies between $(1, 10^2, 10^3, 10^4)$, resulting in four density ratios. Three mesh resolutions $N \in (16, 32, 64)$ are tested. For each mesh resolution N , the domain is discretized

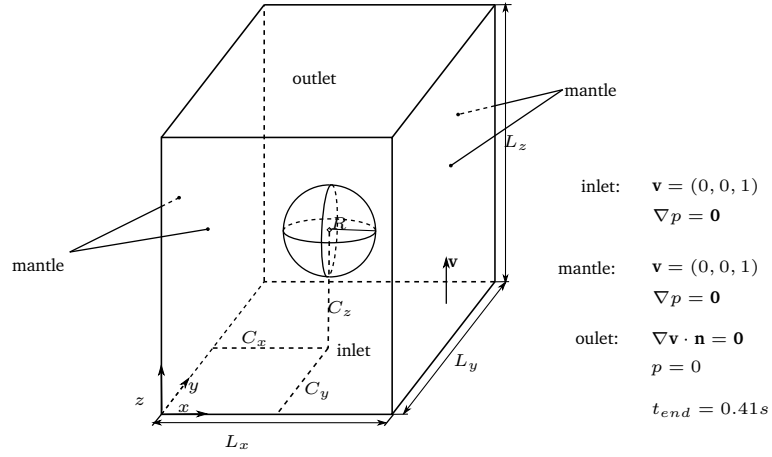


Figure 14: Translating droplet case setup.

		Parameters range			
	Momentum equation	Density ratio	Resolution	Kinematic viscosity	Surface tension coefficient
Group 1	$\partial_t(\rho\mathbf{v}) + \nabla \cdot (\rho\mathbf{v} \otimes \mathbf{v}) = -\nabla p$	$(1, 10^2, 10^3, 10^4)$	(16, 32, 64)	0	0
Group 2	$\partial_t(\rho\mathbf{v}) + \nabla \cdot (\rho\mathbf{v} \otimes \mathbf{v}) = -\nabla p - (\mathbf{g} \cdot \mathbf{x})\nabla\rho$ $+ \nabla \cdot \mu (\nabla\mathbf{v} + (\nabla\mathbf{v})^T) + \mathbf{f}_\Sigma$	$(1, 10^2, 10^3, 10^4)$	(16, 32, 64)	(0.057735, 0.018257, 0.0057735, 0.0)	1

Table 1: The parameters range of the case group 1 and the case group 2.

equidistantly into $1.2N^3$ hexahedral cells, as shown in fig. 15. The exact solution is given by $\mathbf{v}_c^{n+1} = \mathbf{v}_c^n = \mathbf{v}_c(t_0) = \mathbf{v}_a$ and can be used to verify the numerically consistent discretization of the single-field conservative two-phase momentum convection.

Viscosity and surface tension forces are included in the second test case group. The same range of density ratios is simulated, $\rho^-/\rho^+ \in (1, 10^2, 10^3, 10^4)$. The same kinematic viscosity is used for the ambient and the droplet phase, namely $\nu \in (0.057735, 0.018257, 0.0057735, 0.0)$.

The surface tension coefficient is constant $\sigma = 1$.

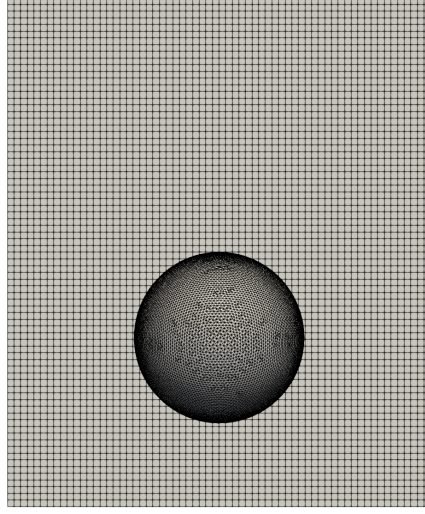


Figure 15: Half section of mesh $N = 64$, droplet at initial position.

6.2.1 Droplet translation without viscosity and surface tension forces

When the momentum is transported only by advection, no forces are exerted on the droplet body and surface. As a result, the velocity field in the overall domain should remain spatially constant and equal to $\mathbf{v}_a = (0, 0, 1)$. The maximum norm L_∞ is employed to measure how much the numerical velocity deviates from the analytical one, i.e.,

$$L_\infty(\mathbf{v}) = \max_i \left(\frac{\|\mathbf{v}_i - \mathbf{v}_a\|}{\|\mathbf{v}_a\|} \right), \quad (6.4)$$

where \mathbf{v}_i denotes velocity of all cells. The previous SAAMPLE method [2] can cause large nonphysical interface deformations leading to a complete deterioration of the solution, visible for a verification configuration in the left image in fig. 17. The deterioration is amplified by the $p - \mathbf{v}$ coupling algorithm that will calculate a pressure field p that enforces $\nabla \cdot \mathbf{v} = 0$. This, in turn, causes artificial acceleration in all cells where $\mathbf{v}_c^{n+1} \neq \mathbf{v}_a$. The consistent ρ LENT method ensures the shape of the droplet is preserved, as shown on the right image in fig. 17.

The fig. 16a contains the velocity error calculated with the old, inconsistent method. Every line in the diagram is labeled by the number of the case, mesh resolution N , and droplet

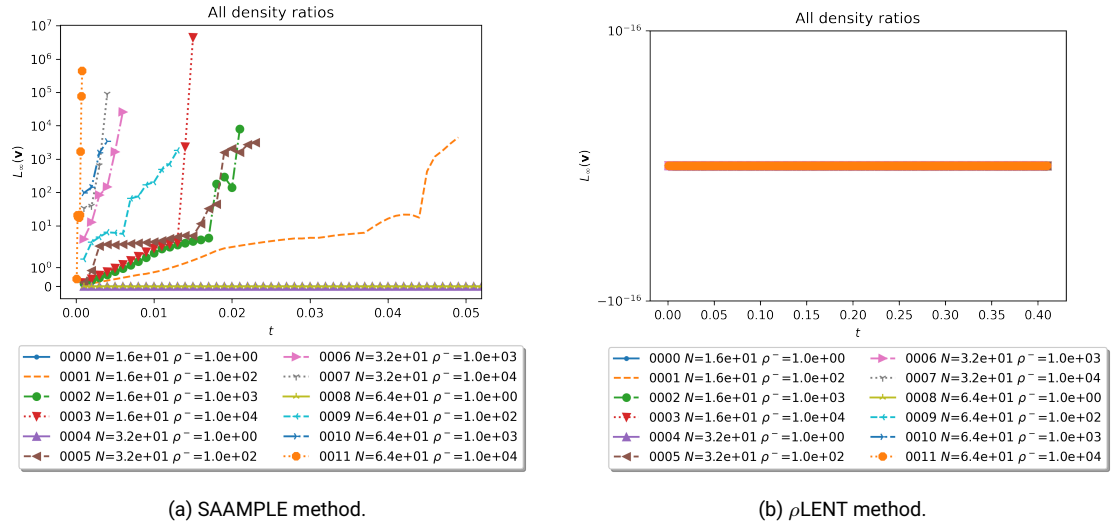


Figure 16: Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$: the left figure depicts the results from SAAMPLE algorithm, the right shows the results from ρ LENT method.

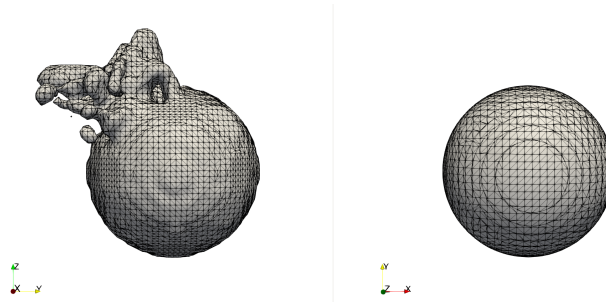


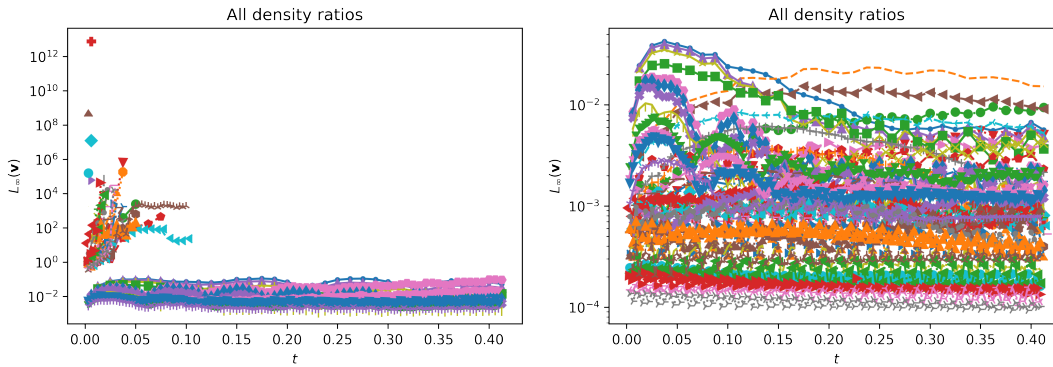
Figure 17: Comparison of the strong interface deformation with SAAMPLE method (left) and the numerically consistent interface shape of the ρ LENT method. Parameters: $N = 64$, $\rho^-/\rho^+ = 10^4$, $t = 0.0008s$.

density ρ^- . The default ambient density is 1. Thus, the ρ^- also represents the density ratio. As shown in fig. 16a, all cases with a density ratio higher than 1, namely $\rho^- > 1$, diverge and stop at early stage. Cases with a very high density ratio of 10^4 (e.g., case 0011 and 0003) fail

catastrophically. The complete results are shown in fig. 48, in appendix 1.

When ρ LENT is used, as shown in fig. 16b, the velocity error remains exactly 0 in all cases. This means that the interface velocity remains consistent with the ambient flow and is unaffected by the mesh resolution and density ratio. The results demonstrate the exact recovery of numerical consistency for the advection of the two-phase momentum, using an implicitly discretized momentum term in a conservative formulation of single-field two-phase Navier-Stokes equations.

6.2.2 Droplet translation with viscosity and surface tension forces



(a) SAAMPLE: interface stable only for cases with density ratio $\rho^-/\rho^+ = 1$ (b) ρ LENT: interface stable for density ratios $\rho^-/\rho^+ \in (1, 10^2, 10^3, 10^4)$

Figure 18: Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ for the viscous flow with surface tension forces: the left diagram depicts the results from the SAAMPLE method, and the right diagram contains the results from the ρ LENT method. The legends of these diagrams are large, and the full information is available in Appendix : fig. 49 for fig. 18a, fig. 50 for fig. 18b.

Here, viscous and capillary forces are taken into account when solving the momentum equation. Since SAAMPLE is a well-balanced algorithm [2] - SAAMPLE balances the discrete surface tension force exactly with the pressure gradient when constant curvature is used,

using the same discretization for the pressure gradient and the surface-normal gradient of the volume fraction [98]. The force-balance is maintained also if the curvature is exactly calculated and propagated as a constant in the interface-normal direction. For numerically approximated curvature, the balance is obtained on a dissipation timescale with respect to initial perturbations. The translating droplet test case combines the force-balance requirement in the droplet’s frame of reference, with the requirement for numerical consistency of the two-phase momentum advection. In the absence of gravity, such a droplet does not accelerate or decelerate. The temporal evolution of L_∞ is shown in fig. 18. The inconsistent method remains stable only for $\rho^-/\rho^+ = 1$. For the results of all other cases, i.e., with $\rho^-/\rho^+ > 1$, the velocity error increases exponentially, and the simulations crash. In contrast, as depicted in fig. 18b, the ρ LENT demonstrates numerically stable results for all tested density ratios. Additional numerical errors are introduced compared with two-phase momentum advection, specifically when approximating the curvature [2]. The approximation of curvature in [2] recovers accurate L_2 norms of the curvature errors for a sphere, in the range $[10^{-4}, 10^{-3}]$ for discretization lengths in the range $[128^{-1}, 16^{-1}]$ in the unit-box solution domain. Because of the numerically approximated curvature, L_∞ cannot exactly be equal to zero, as shown in fig. 16b. However, as seen in fig. 18b, the final L_∞ error given by eq. (6.4) 10^{-4} and 10^{-2} , which is acceptable.

6.2.3 Translating sub-millimeter droplet with realistic physical properties

materials/properties (25 °C)	density (kg m ⁻³)	kinematic viscosity (m ² s ⁻¹)	surface tension (N m ⁻¹)	density ratio
air	1.1839	1.562×10^{-5}	---	--- [99]
water	997.05	8.926×10^{-7}	0.07213 (in air)	842.17 (in air) [99]
mercury	13.5336×10^3	1.133×10^{-7}	0.4855 (in air)	11431.37(in air) [99]
silicone oil (cSt 10)	0.934×10^3	1.088×10^{-5}	0.0201 (in air)	788.92(in air) [100]
silicone oil (cSt 50)	0.96×10^3	5×10^{-5}	0.032 (in water)	0.96 (in water) [101]

Table 2: Realistic fluid properties are combined into four tests: water droplet/air ambient, mercury droplet/air ambient, silicone oil droplet/air ambient, silicone oil droplet/water ambient.

The physical properties including densities, viscosities, and surface tension coefficients in the widely used translated droplet case from Popinet [98] are not related to physical two-phase flows systems. In this test, the case has been adapted, and small droplet dimensions have been used to challenge the method in terms of surface tension force approximation for capillary problems, along with the utilization of real-world fluid pairings with challenging density ratios.

Table 2 contains the physical properties used for the test-case configuration of the translating sub-millimeter droplet with realistic physical properties. In terms of size, a spherical droplet of radius $R = 0.25$ mm is translating a distance of three diameters with velocity 0.01 m/s in z -direction of the rectangular solution domain ($L_x = L_y = 5R, L_z = 10R$). The initial centroid position of the droplet is $(2.5R, 2.5R, 2R)$. Surface tension and viscous forces are not considered for this setup.

As depicted in fig. 19, it is obvious that $L_\infty(\mathbf{v})$ remains stable over time when the droplet translates. Even in the cases with a density ratio of over 10^4 , as shown in fig. 19d, no matter how high the resolution is, the results from ρ LENT the method can reach machine precision.

The fig. 20 illustrates the results of the same realistic droplets' cases, considering the influence of viscous forces and surface tension. It is observed that the errors decrease as the resolution increases, reaching magnitudes as low as 10^{-5} for all cases. This indicates the excellent capability of ρ LENT to handle a wide range of density ratios in such cases.

Apart from the observation mentioned above, table 3 reveals another advantage of ρ LENT method - high computational efficiency. As shown in table 3, the ρ LENT method demonstrates very high computational efficiency in serial. Increasing the parallel computational efficiency requires further research, specifically, regarding a more efficient message-passing parallel implementation for unstructured Level Set / Front Tracking.

6.3 Oscillating droplet

An ellipsoidal droplet is submerged in an ambient fluid with an approximate axially symmetric solution provided by Lamb [102]. The solution can be represented as a summation of a

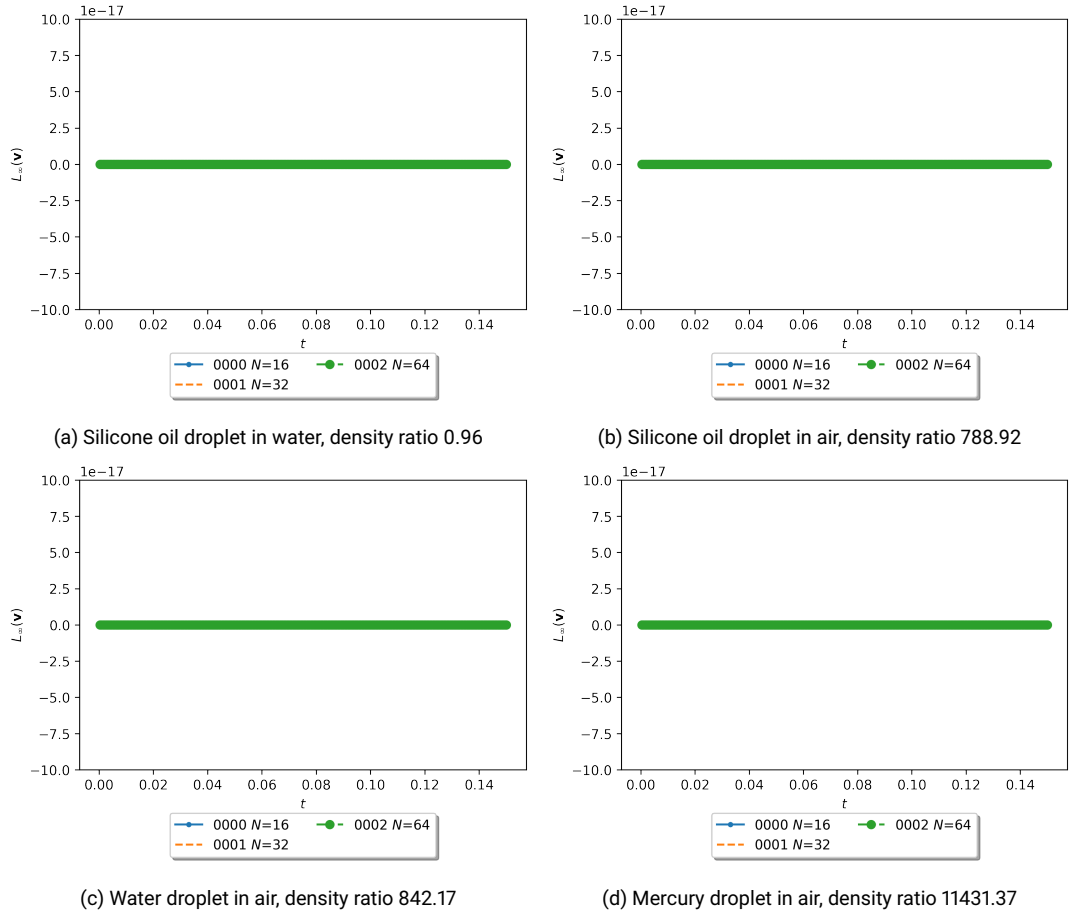


Figure 19: Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ with pure advection: ρ LENT method used in simulating two-phase flows with different density ratios, mesh resolution: $N \in (16, 32, 64)$.

constant and a Legendre polynomial $P_n(\cos \theta)$, i.e.,

$$R(\theta, t) \doteq R_0 + a_n P_n(\cos \theta) \sin(\omega_n t), \quad \theta \in [0, 2\pi], \quad (6.5)$$

where R_0 is the initial unperturbed radius, a_n is the amplitude of the n -th oscillation mode, θ is the angle between the radius line of a droplet point and the symmetric axis, ω_n represents

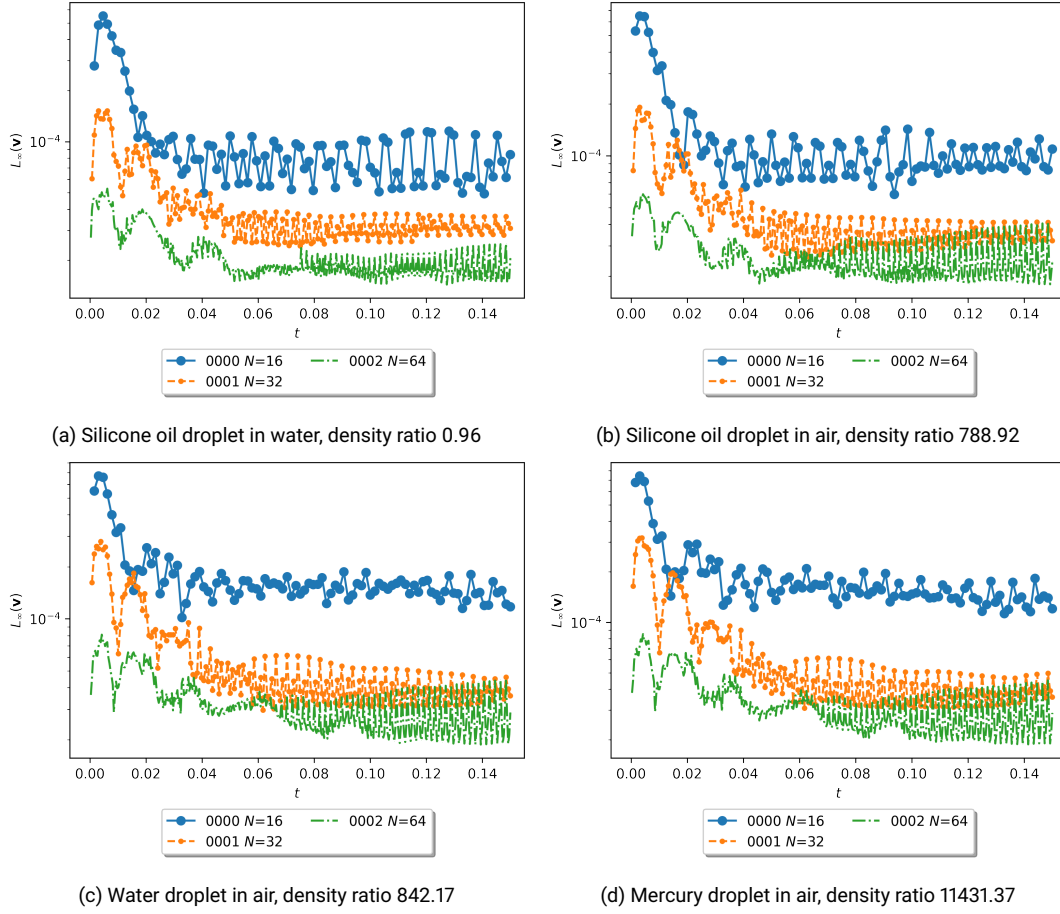


Figure 20: Temporal evolution of velocity error norm $L_\infty(\mathbf{v})$ with the effect of viscosity and surface tension: ρ LENT method used in simulating two-phase flows with different density ratios, mesh resolution: $N = 16, 32, 64$.

the oscillation frequency. The latter ω_n has the form

$$\omega_n^2 = \frac{n(n+1)(n-1)(n+2)\sigma}{[(n+1)\rho_d + n\rho_a]R_0^3}, \quad (6.6)$$

where n is the mode number, ρ_d, ρ_a represent the droplet and ambient flow density respectively, σ indicates the surface tension coefficient. Lamb [102] derived eq. (6.5) neglecting the viscous

cases	resolution	serial execution time (s)
silicone oil droplet / water	16	6.62
	32	74.34
	64	929.79
water droplet / air	16	7.77
	32	91.64
	64	1324.38
mercury droplet / air	16	7.71
	32	94.57
	64	1334.36
silicone oil droplet / air	16	7.31
	32	83.43
	64	1318.43

Table 3: Serial execution time for the ρ LENT method.

effect, and used constant a_n . Chandrasekhar [103], Miller and Scriven [104], and Prosperetti [105] extended the expression of a_n to include the influence of the viscosity, where the amplitude a_n decays exponentially over time by

$$a_n(t) = a_0 e^{-\gamma t}, \quad \gamma = \frac{(n-1)(2n+1)\nu}{R_0^2}, \quad (6.7)$$

where ν is the kinematic viscosity. Hiller and Kowalewski [106] conducted a series of experiments to validate the decay expression.

The physical properties of mercury and air from 2 is applied to the droplet and the ambient fluid. The droplet interface is initialized with the parameters: $R_0 = 0.01$, $n = 2$, $\epsilon = 0.00025$, $t = \pi/(2\omega_n)$ and the center $(0.0200001, 0.0199999, 0.020000341)$. The computational domain size is $(0, 0, 0) \times (0.04, 0.04, 0.04)$. The gravity is neglected in this case. At the simulation's beginning, the droplet and the flow are still, i.e. $\mathbf{v}(t=0) = 0$ holds for the whole field. Since the analytical frequency ω_a can be acquired from eq. (6.6), the results are evaluated by an error norm

$$L_1(\omega) = \frac{|\omega - \omega_a|}{\omega_a}. \quad (6.8)$$

Three mesh resolutions are tested to verify the convergence of both methods. The results of the SAAMPLE[2] and ρ LENT with increasing mesh resolutions are summarized in the table 4.

As shown in fig. 21, the oscillating frequencies calculated from SAAMPLE do not converge. On the contrary, when deploying the new consistent method, L_1 norm exhibits second-order convergence.

Grid size h	Background mesh		Front mesh		SAAMPLE		ρ LENT	
	Points	Cells	Points	Tris.	Frequency	L_1 norm	Frequency	L_1 norm
0.0016	17576	15625	3150	6296	16.255923	0.040397	16.346002	0.035080
0.0008	132651	125000	12662	25320	16.444685	0.029254	16.888921	0.003031
0.0004	1030301	1000000	50732	101460	16.218606	0.042600	16.928274	0.000708

Table 4: The analysis of the oscillation frequency convergence and its comparison between the SAAMPLE method and the consistent ρ LENT method.

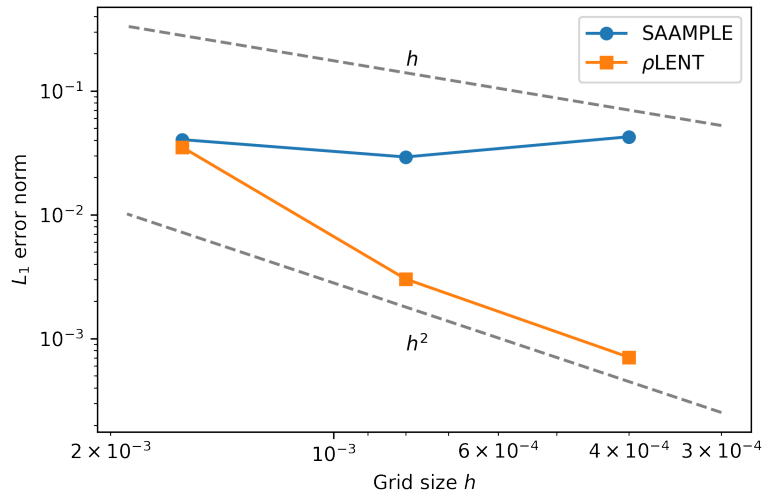


Figure 21: L_1 norms of the oscillation frequency errors.

6.4 Rising bubble

In this test case, the proposed method is applied to a single bubble rising in quiescent viscous liquid. The configuration from Anjos et al. [5] is adopted, who simplified the rising bubble

experiments previously conducted by Bhaga and Weber [3] and selected three different viscosity ratios to perform tests and comparisons. In this section, the focus is on the most challenging case, i.e. the case with the smallest viscosity ratio, which corresponds Morton number $Mo = g\nu_l^4/\rho_l\sigma^3 = 1.31$, where g is the gravitational acceleration value, and ν_l , ρ_l , σ indicate the viscosity, density of the ambient liquid and the surface tension. The initial state of the air bubble is idealized to be spherical with a diameter of $D = 2.61$ cm. The physical properties of the air are characterized by a viscosity of 1.78×10^{-5} kg/ms and a density of 1.225 kg/m³, whereas the properties of the liquid are defined by a viscosity of 0.54 kg/ms and a density of 1350 kg/m³. Additionally, the surface tension between the air bubble and the liquid is 0.078 N/m. The computational domain is defined as $(-4D, -4D, -2D) \times (4D, 4D, 6D)$, which show the positions of space diagonal vertices of the computational domain, and the initial position of the bubble is set as the origin, $(0, 0, 0)$. As the parallel computing module

time=0.07

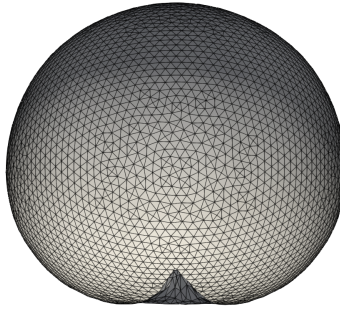


Figure 22: SAAMPLE method: the collapsed bubble shape caused by numerical inconsistency.

in the ρ LENT method is still in the developmental phase, the entire domain was resolved using a single core. Consequently, relatively coarse meshes were utilized to simulate the motion of the bubble, i.e. $N \in (64, 96, 128, 160)$, where N indicates the grid numbers in all three directions of the computational domain. To estimate the results, a set of dimensionless

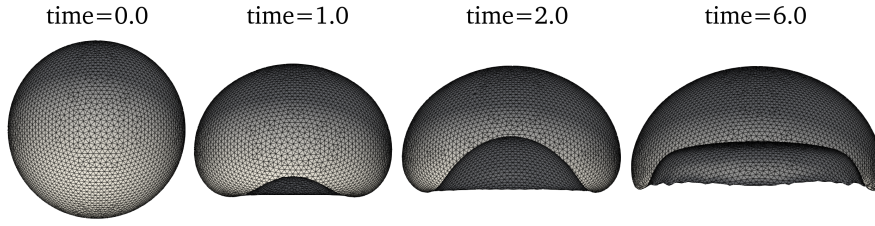


Figure 23: ρ LENT: the temporal evolution of bubble's shape with resolution $N = 128$.

characteristic variables was introduced as follows:

$$\mathbf{w} = \frac{\mathbf{v}}{\sqrt{gD}}, \quad t = \sqrt{\frac{g}{D}}\tau, \quad (6.9)$$

where τ indicates the realistic time. When deploying the inconsistent method, the simulation crashed at an early stage, as shown in 22. Inconsistent method deployment resulted in a simulation crash at an early stage. The velocity of the bubble's bottom region increased abruptly, causing the front's vertices in that region to have much higher velocity than the neighbor region, which explains the bottom sharp cone formation as shown in Figure 22. Conversely, Figure 23 depicts the temporal evolution of the bubble's shape using the consistent method with $N = 128$. The predicted bubble shapes show good agreement with the previous simulation results [4, 5, 92] and the experimental results [3]. Additionally, 24 shows a comparison of the bubble's rising velocities between the ρ LENT method and some previous works. At the acceleration stage, the predicted rising velocity from ρ LENT method with the finest mesh $N = 160$ agrees remarkably well with the results from Anjos et al. [5], whereas the velocities from the cases with coarser meshes are slightly higher than the results from Anjos et al. [5]. The deceleration stage of the bubble can be observed for all cases with different resolutions, which also exists in the results of Anjos et al. [5]. Except for in the case with the coarse mesh $N = 64$, the rising velocities in cases with higher resolutions reach a stable state and converge to the experimental value from Bhaga and Weber [3].

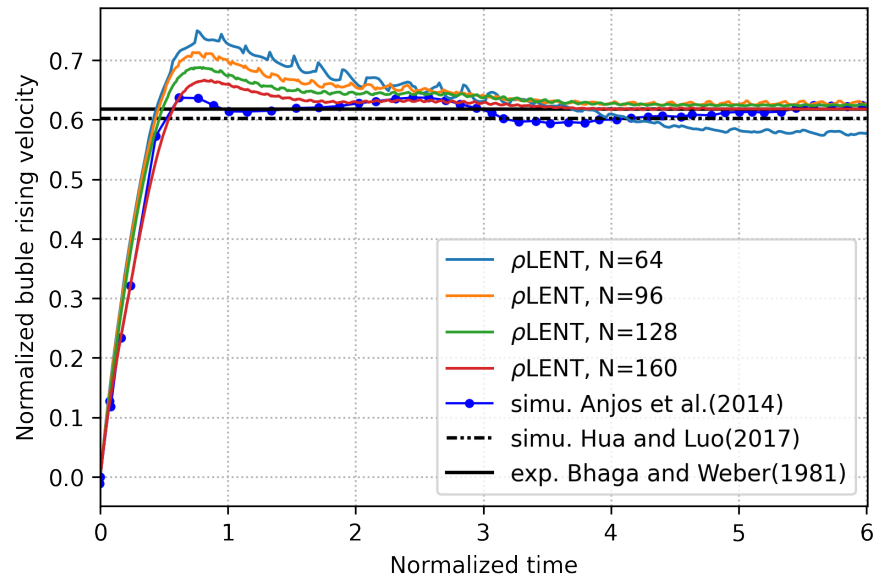


Figure 24: The average bubble velocities from ρ LENT with four resolutions are compared with: the experimental results (black solid line) from Bhaga and Weber [3], the simulation results (black dashed double-dotted line) from Hua and Lou [4], and the extracted simulation results (blue solid dotted line) from Anjos et al. [5]. Simu. and exp. are the abbreviation of simulation and experiment.

Inconsistencies in Unstructured Geometric Volume-of-Fluid Methods for Two-Phase Flows with High Density Ratios

7 Introduction

Geometric flux-based VOF methods [34] are widely regarded as being consistent in handling two-phase flows with high density ratios. However, although the conservation of mass and momentum is deemed consistent for two-phase incompressible single-field Navier-Stokes equations without phase change, as discussed in section 3.5.4 and [47], small inconsistencies may easily be introduced by discretization, resulting in very large errors or catastrophic failure. The consistency conditions derived for ρ LENT in section 3.5.4 are applied to flux-based geometric VOF methods [34], and the consistent discretization is implemented into the plicRDF-isoAdvect vector geometrical VOF method [36]. It is found in this chapter that the equivalence between the scaled volume fraction equation and the mass conservation equation is destroyed, depending on the choice of temporal and convective term discretization schemes, when computing the mass flux by scaling the geometrically computed fluxed phase-specific volume. Two solutions are proposed in this study. First, based on the analysis of discretization errors, a consistent combination of the temporal discretization scheme and the interpolation scheme for the momentum convection term should be utilized. Second, similar to section 3.5.4, an auxiliary mass conservation equation is solved with a geometrical calculation of the face-centered density. The equivalence between these two approaches is mathematically proved,

and their numerical stability for density ratios within $[1,10^6]$ and viscosity ratios within $[10^2,10^5]$ is verified and validated in this chapter.

In this chapter, the consistency requirements imposed by the single-field Navier-Stokes equations derived in section 3.5.4 are applied to flux-based geometric VOF methods [34], using the isoAdvector method [6, 36, 49] to verify and validate these findings. The consistency requirements from last chapter indicate that geometric flux-based VOF methods are inherently consistent in handling high density ratios. On the modeling level, an exact solution of the volume fraction equation is equivalent to solving the mass conservation equation. On the discrete level, mass fluxes needed in the implicitly discretized single-field momentum convection term are consistently computed by flux-based VOF methods, by scaling the fluxed phase-specific volume. However, it is demonstrated that inconsistencies can easily arise through the choice of an inconsistent combination of the temporal integration scheme and the interpolation scheme for the two-phase momentum conservation term, leading to significant errors for small density ratios and catastrophic failures for large density ratios.

The review is provided of the single-field formulation of incompressible two-phase Navier-Stokes equations, and their discretization is analyzed using the collocated unstructured Finite Volume method, in section 3. In section 8.1, it is shown that a consistent combination of the temporal discretization scheme and the interpolation scheme for the momentum convection term is necessary to ensure stable solutions with high density ratios. In sections 8.2 and 8.3, the introduction of an auxiliary mass conservation equation with a geometrical calculation of the face-centered density to the geometrical VOF method isoAdvector [6, 36, 49] is discussed. The equivalence between these two approaches is proved, and their numerical stability for density ratios of $[1,10^6]$ and viscosity ratios of $[10^2,10^5]$ is verified and validated in the results section 9.

Literature review

A detailed review of two-phase flow simulation methods that address the challenges in handling high density ratios is provided in section 3.5.4, and here more current literature contributions are listed.

Huang, Lin, and Ardekani [107] proposed the mixed Upwind/Central Weighted Essentially Non-Oscillatory (WENO) scheme on a staggered structured grid, an extension of the conventional WENO scheme [108] on collocated grids, to spatially discretize the nonlinear convective term in conservative form. The Upwind WENO scheme is used to evaluate velocity at cell faces, while 3 different forms of Central WENO scheme are applied to evaluate mass flux in x-/y-directions as well as density at cell corners. In addition to the mentioned spatial discretization scheme, Huang, Lin, and Ardekani [107] proposed also a semi-implicit projection scheme to decouple pressure and velocity in momentum transport equation. A backward difference scheme combined with a special treatment for the viscous term is introduced to discretize the momentum equation without the pressure gradient. The viscous term containing the intermediate velocity is split into two parts, one of which comprises the constant arithmetic mean of two phases' viscosity and the intermediate velocity, another one comprises the updated viscosity and the explicit velocity (referring to [109, Appendix A]). Numerous 2D cases with density ratio [1, 1000] are tested, whose results are in good agreement with analytical and experimental results. The interface capturing method in [107] is selected to phase-field. The authors indicated that the mixed Upwind/Central WENO scheme is able to be coupled with any interface capturing/tracking method.

Xie et al. [110] introduced a consistent and balanced-force model with Consistent Balanced-force Level Set and Volume Of Fluid (CBLSVOF) on polyhedral unstructured grids. A hybrid algebraic/geometric VOF method based on Tangent of Hyperbola Interface Capturing with Quadratic surface representation and Gaussian Quadrature (THINC/QQ) [111] is developed to capture interface, while the level set function constructed from volume fraction is exploited to improve the accuracy of interface curvature evaluation. To achieve the consistency, the convective term of both the volume-fraction equation and momentum equation in the conservative form are discretized in the same manner, wherein the velocity and volume fraction are reconstructed by using the identical high-order reconstruction scheme based

on a quadratic polynomial function. The balanced-force formulation proposed firstly by Francois et al. [112] is adapted in this work to eliminate the parasitic current at the interface. The interface curvature estimated by the continuous signed-distance function instead of the abruptly changing volume fraction function offers computational simplicity and numerical stability. A 3D single bubble, and two bubbles coalescence cases with the density ratio of magnitude 10^3 are tested. The results agree well with previous experimental results.

Desmons and Coquerelle [113] proposed a generalized approach known as the High-Order Momentum Preserving (HOMP) method. A high-order temporal and spatial scheme, i.e., Runge-Kutta (RK)2 and WENO5,3 [114], are used to discretize an auxiliary advection equation of characteristic function χ that is compatible with the mass equation and independent of the underlying interface-representation function. The momentum equation is then discretized using the same high-order schemes as before, and the density is deduced from the characteristic function χ , which is computed from the auxiliary advection equation rather than from the interface transport step. During the discretization, both the auxiliary advection equation and the momentum equation remain in the conservative form. The authors combined several interface representation methods, i.e., Volume Of Fluid, Level Set Method, and Moment Of Fluid, with HOMP and then tested them. The results from various selected validation cases, including water and air, demonstrate good agreement with the literature results. It is worth noting that no theoretical explanation is offered as to why the auxiliary advection equation is required, rather than just maintaining the consistency between the interface transport equation and the momentum equation. Alternatively, the higher-order WENO scheme requires very large stencils of variable width when used with the finite-volume method, making its parallel implementation inefficient using the domain-decomposition / message-passing parallel programming model.

El Ouafa, Vincent, and Le Chenadec [115] devised a fully coupled solver for simulating incompressible two-phase flows characterized by large density and viscosity ratios on a staggered structured mesh. The interface is captured by a Piecewise Linear Interface Calculation (PLIC)-Volume of Fluid (VOF) method. In this solver, the linearized momentum and continuity equations arising from the implicit solution of the fluid velocities and pressure are solved simultaneously. To complete the one-fluid formulation of the momentum equation, velocity boundary conditions of the fluid domain are imposed by adding a penalty term. The authors

interpreted that the errors from the splitting operator and the additional pressure boundary conditions for segregated methods are sensitive to density ratio. Using the fully coupled method can overcome the problems. The consistency between the advection equation of volume fraction and the momentum equation is not necessary for this solver. A few cases featuring density ratios up to 10^6 and viscosity ratios up to 10^{10} are tested, whose results show remarkable stability and accuracy.

Yang, Lu, and Wang [116] introduced a robust methodology that integrates the consistent mass-momentum convection approach from Nangia et al. [52] with the CLSVOF interface capturing method by Sussman and Puckett [62] to tackle the challenge posed by high-density ratio problems encountered in high-Reynolds-number flows. At each time step, the interface evolution is initiated using the CLSVOF method to provide an initial value for the mass equation. Subsequently, the conservative form of the momentum equation and mass equation are simultaneously solved employing consistent temporal (second-order RK) and spatial (third-order CUI) discretization schemes at all cell faces of the staggered structured mesh. The authors emphasized the importance of employing identical densities in the discretized mass and momentum equations at the two substeps of the second-order RK method to ensure robust simulation of high-Reynolds-number two-fluid flows with high density ratios. The proposed method yields accurate predictions for both two-dimensional and three-dimensional wave breaking cases with a density ratio of 10^3 and Reynolds number of 10^8 .

Li, Liu, Wan, et al. [117] proposed a simple and robust method to simulate high density ratio interfacial flows. Similar to the method in [116], the auxiliary mass equation is solved together with the momentum equation. However, instead of using the consistent discretization schemes in space and time, this method applies the identical cell-center velocity updated by solving the mass equation and the mass flux to the discretized momentum equation to maintain consistency between them. Three different schemes of VOF, i.e, PLIC-VOF, the spatial filtering VOF, and the Tangent of Hyperbola Interface Capturing with Quadratic surface representation (THINC)-VOF, are implemented in the work to transport the interface. The improvements of stability and accuracy for all three schemes in canonical cases like heavy droplet advection and falling droplet show the generality of the consistent method to deal with high density ratios problem.

Zeng et al. [118] transferred the consistent treatment of conservative mass and momen-

tum equations on staggered Cartesian grids introduced by Nangia et al. [52] to multilevel collocated grids. The adapted level set method with a multilevel reinitialization technique is applied to capture the interface. The consistent scheme achieves a numerically stable and reasonably accurate solution to realistic multiphase flows, such as breaking waves with a high Reynolds number.

8 Methodology review

The single-field formulation of incompressible two-phase Navier-Stokes Equations (NSE) without phase change (eq. (2.15)) is especially relevant for engineering applications because it provides a solid modeling basis for two-phase flows with fluid interfaces that can arbitrarily deform, breakup and merge, provided that the method responsible for advecting the phase indicator does not impose its own restrictions regarding fluid interface deformation and topological changes. Single-field NSE also embed strict consistency requirements section 3.5.4: an equivalence between the conservation of mass and the phase-indicator (volume) conservation. These requirements translate on the discrete level into requirements for the computation of the mass flux in the discretized two-phase momentum convection term in eq. (2.15), analyzed in the following sub-section.

To preserve a continuous derivation and avoid jump back and forth frequently to check equations, some important equations shown in section 5 are replicated in this section.

8.1 Single-field mass conservation and volume fraction conservation

The discretization of the single-field two-phase momentum convection term from eq. (2.15) is reformulated by

$$\int_{\Omega_c} \nabla \cdot (\rho \mathbf{v} \mathbf{v}) dV = \int_{\partial \Omega_c} (\rho \mathbf{v} \mathbf{v}) \cdot \mathbf{n} ds = \sum_{f \in F_c} \rho_f F_f \mathbf{v}_f + e_{\rho \mathbf{v}, con}(h^2), \quad (8.1)$$

with the face-centered volumetric flux

$$F_f := \mathbf{v}_f \cdot \mathbf{S}_f. \quad (8.2)$$

To compute $\rho_f F_f$, the volume fraction is defined using the phase-indicator eq. (2.9) as

$$\alpha_c(t) := \frac{1}{|\Omega_c|} \int_{\Omega_c} \chi(\mathbf{x}, t) dV. \quad (8.3)$$

The volume fraction advection equation [34]

$$\frac{d}{dt} \int_{\Omega_c} \chi dV = |\Omega_c| \frac{d}{dt} \alpha_c(t) = - \int_{\partial\Omega_c} \chi \mathbf{v} \cdot \mathbf{n} dS, \quad (8.4)$$

is equivalent to the advection equation for the phase $\Omega^+(t)$ indicated by $1 - \chi(\mathbf{x}, t)$, namely

$$\frac{d}{dt} \int_{\Omega_c} (1 - \chi) dV = -|\Omega_c| \frac{d}{dt} \alpha_c(t) = - \int_{\partial\Omega_c} (1 - \chi) \mathbf{v} \cdot \mathbf{n} dS, \quad (8.5)$$

since $\frac{d}{dt} \int_{\Omega_c} dV = 0$, as $\Omega_c \neq \Omega_c(t)$, and $\int_{\partial\Omega_c} \mathbf{v} \cdot \mathbf{n} dS = \int_{\Omega_c} \nabla \cdot \mathbf{v} dV = 0$ for incompressible two-phase flows without phase change.

The mass conservation with the single-field density eq. (2.10) in a fixed control volume Ω_c states

$$\frac{d}{dt} \int_{\Omega_c} \rho dV = - \int_{\partial\Omega_c} \rho \mathbf{v} \cdot \mathbf{n} dS. \quad (8.6)$$

Inserting eq. (2.10) into the r.h.s. of eq. (8.6) and integrating over the time step $[t^n, t^{n+1}]$ results in

$$= \rho_c^n + \frac{1}{|\Omega_c|} \left[-\rho^- \int_{t^n}^{t^{n+1}} \sum_{f \in F_c} \int_{S_f} \chi \mathbf{v} \cdot \mathbf{n} dS dt - \rho^+ \int_{t^n}^{t^{n+1}} \sum_{f \in F_c} \int_{S_f} (1 - \chi) \mathbf{v} \cdot \mathbf{n} dS dt, \right] \quad (8.7)$$

Inserting the eqs. (8.4) and (8.5) in the eq. (8.7) to replace the sums of surface integrals results in

$$\rho_c^{n+1} = \rho_c^n + \frac{\rho^-}{|\Omega_c|} |\Omega_c| \int_{t^n}^{t^{n+1}} \frac{d}{dt} \alpha_c(t) dt + \frac{\rho^+}{|\Omega_c|} |\Omega_c| \int_{t^n}^{t^{n+1}} -\frac{d}{dt} \alpha_c(t) dt, \quad (8.8)$$

leading finally to

$$\rho_c^{n+1} = \rho_c^n + (\rho^- - \rho^+) (\alpha_c^{n+1} - \alpha_c^n). \quad (8.9)$$

Equation (8.9) shows that solving eq. (8.6) for ρ_c^{n+1} is equivalent to adding the solution of the volume fraction equation scaled with $(\rho^- - \rho^+)$ to ρ_c^n .

Alternatively, integrating the single-field density model (eq. (2.10)) over the control volume Ω_c gives

$$\int_{\Omega_c} \rho dV = \int_{\Omega_c} \rho^- \chi + \rho^+ (1 - \chi) dV. \quad (8.10)$$

Dividing eq. (8.10) by $|\Omega_c|$ using eq. (8.3) results in the discrete single-field density model,

$$\rho_c(t) = \rho^- \alpha_c(t) + \rho^+ (1 - \alpha_c(t)), \quad (8.11)$$

which, evaluated at t^{n+1} and t^n and subsequently subtracted, results in

$$\begin{aligned} \rho_c^{n+1} - \rho_c^n &= \rho^- \alpha_c^{n+1} + \rho^+ (1 - \alpha_c^{n+1}) - \rho^- \alpha_c^n - \rho^+ (1 - \alpha_c^n) \\ &= (\rho^- - \rho^+) (\alpha_c^{n+1} - \alpha_c^n), \end{aligned}$$

which is eq. (8.9): solving the volume fraction advection equation eq. (8.4) and using the single-field density model to compute the cell centered density by eq. (2.10) is equivalent to solving the mass conservation equation (eq. (8.6)).

Note that all equations eq. (8.3) - eq. (8.11) are exact, as $\partial\Omega_c := \bigcup_{f \in F_c} S_f$, where S_f are non-linear surfaces that bound the control volume Ω_c , and reformulating the integration in time is exact by the fundamental theorem of calculus.

Ghods and Herrmann [51] introduce the auxiliary mass conservation equation as a means for ensuring the consistency of the two-phase momentum convection, and the theoretical reasoning for the auxiliary mass conservation equation is provided in section 3.5.4. Contrary to Ghods and Herrmann [51], eq. (8.9) and eq. (8.10) both demonstrate, that the solution of the volume fraction equation in the context of the VOF method [34] is exactly equivalent to the solution of the mass conservation equation, rendering an auxiliary mass conservation equation unnecessary.

However, the following question arises: if flux-based algebraic/geometric VOF methods inherently ensure numerical stability for the two-phase momentum convection with high

density ratios, where do the numerical inconsistencies reported throughout the literature [7, 76, 119–122] come from?

Although eqs. (8.9) and (8.10) show the inherent consistency of VOF methods in the mathematical model, the discrete computation of α_c^{n+1} , the approximation of the mass flux $\rho_f F_f$, the choice of the temporal scheme and the flux limiting scheme, can potentially cause inconsistencies.

The explication is started with the discrete computation of α_c^{n+1} and the approximation of $\rho_f F_f$ by scaling the fluxed phase-specific volume. For a second-order accurate flux-based VOF method, the approximations applied to the temporal and surface integrals for the fluxed phase-specific volumes V_f^α when solving eq. (8.4) lead to

$$\alpha_c^{n+1} = \alpha_c^n - \frac{1}{|\Omega_c|} \sum_{f \in F_c} \int_{t^n}^{t^{n+1}} \int_{S_f} \chi \mathbf{v} \cdot \mathbf{n} dS dt = \alpha_c^n - \frac{1}{|\Omega_c|} \sum_{f \in F_c} |V_f^\alpha|_s + e_{\alpha_t}(\Delta t^p) + e_{\alpha_h}(h^2), \quad (8.12)$$

with $e_{\alpha_t}(\Delta t^p)$ and $e_{\alpha_h}(h^2)$ as temporal and spatial volume fraction discretization errors. Note that

$$|V_f^\alpha|_s := \text{sgn}(F_f) |V_f^\alpha| \quad (8.13)$$

is a signed magnitude of a *phase-specific volume* V_f^α [34], whose sign is determined by the volumetric flux $\rho_f F_f$.

The phase-specific volume $|V_f^\alpha|_s$ can be used to approximate the mass flux $\rho_f F_f$ using eq. (8.9) and eq. (8.12). Reordering eq. (8.12), results in

$$\alpha_c^{n+1} - \alpha_c^n = \frac{1}{|\Omega_c|} \sum_{f \in F_c} |V_f^\alpha|_s + e_{\alpha_t}(\Delta t^p) + e_{\alpha_h}(h^2) \quad (8.14)$$

Inserting eq. (8.14) into eq. (8.9) results in

$$\rho_c^{n+1} - \rho_c^n = \frac{\rho^- - \rho^+}{|\Omega_c|} \sum_{f \in F_c} |V_f^\alpha|_s + (\rho^- - \rho^+) [e_{\alpha_t}(\Delta t^p) + e_{\alpha_h}(h^2)]. \quad (8.15)$$

Equivalently integrating eq. (8.6) over $[t^n, t^{n+1}]$ results in

$$\rho_c^{n+1} - \rho_c^n = \frac{1}{|\Omega_c|} \sum_{f \in F_c} \int_{t^n}^{t^{n+1}} \rho_f F_f dt = \frac{1}{|\Omega_c|} \sum_{f \in F_c} |M_f|_s + e_{\rho_t}(\Delta t^s) + e_{\rho_h}(h^2), \quad (8.16)$$

with $e_{\rho_t}(t^s), e_{\rho_h}(h^2)$ as the temporal and spatial discretization errors of the mass conservation equation. The right-hand sides of eqs. (8.15) and (8.16) express the mass fluxed through $\partial\Omega_c$ over $[t^n, t^{n+1}]$, i.e. $|M_f|_s$, as the phase-specific volume $|V_f^\alpha|_s$ scaled by the density difference, thus connecting the fluxed mass with the fluxed phase specific volume.

Consistency of the volume fraction advection and the mass conservation on the discrete level requires the equivalence of eqs. (8.15) and (8.16). Equations (8.15) and (8.16) will be exactly the same, only if their errors on the r.h.s. cancel out. Error cancellation is impossible if equations eqs. (8.15) and (8.16) are using different numerical schemes for $|V_f^\alpha|_s$ and $|M_f|_s$, which strongly connects the fluxed mass $|M_f|_s$ with the fluxed phase-specific volume. In other words, if a specific VOF method for $|V_f^\alpha|_s$ is deployed, the fluxed mass from $|V_f^\alpha|_s$ should be computed. This is hypothetical, of course, since there is no need to actually solve two equations that are equivalent. However, it is notable the mass flux is needed for the discretized momentum equation (eq. (2.15)). If the fluxed mass must be computed from the scaled $|V_f^\alpha|_s$ to ensure the consistency of the volume fraction advection and mass conservation, it follows that the mass flux consistent with mass conservation must also be computed from $|V_f^\alpha|_s$, using eq. (8.15).

However, geometrical VOF methods use temporal integration schemes and geometrical algorithms to compute $|V_f^\alpha|_s$ - a time-integrated quantity - that are very unlike the algebraic discretization schemes applied on eq. (2.15). Precisely this difference is a source of very strong instabilities and catastrophic failures for high density ratios.

It helps to consider a concrete example. Consistency of the mass flux can be shown when the mass flux is computed by scaling the fluxed phase-specific volume in eq. (8.15) with $(\rho^- - \rho^+)$, in a simplified first-order geometrical VOF method, which uses the "Euler" temporal integration (rectangle quadrature) of $|V_f^\alpha|_s$, i.e.

$$\begin{aligned}
|V_f^\alpha|_s^{Euler} &= \int_{t^n}^{t^{n+1}} \int_{S_f} \chi \mathbf{v} \cdot \mathbf{n} dS dt = \frac{F_f^n}{|S_f|} \int_{t^n}^{t^{n+1}} \int_{S_f} \chi dS dt + e_{\alpha_t}(\Delta t^2) + e_{\alpha_h}(h^2) \\
&= F_f^n \int_{t^n}^{t^{n+1}} \alpha_f(t) dt + e_{\alpha_t}(\Delta t^2) + e_{\alpha_h}(h^2) \\
&= F_f^n \alpha_f^n \Delta t + e_{\alpha_t}(\Delta t^2) + e_{\alpha_h}(h^2),
\end{aligned} \tag{8.17}$$

where the fraction of the wetted face area $A_f(t)$ is defined as $\alpha_f := A_f(t)/S_f$ in this study,

and $N_t e_{\alpha_t}(\Delta t^2) = \frac{T}{\Delta t} e_{\alpha_t}(\Delta t^2) \propto \Delta t$ results in a first-order temporal quadrature error over the simulated physical time T . Equation (8.17) is a simplified scheme used here only to discuss the mass-flux consistency condition on the discrete level, it is not a practically usable scheme for advecting α_c , because it is significantly less accurate than modern geometrical schemes [34]. Multiplying $|V_f^\alpha|_s^{Euler}$ from eq. (8.17) with $(\rho^- - \rho^+)$ to obtain $|M_f|_s$ ensures the consistency of eqs. (8.15) and (8.16), namely

$$\begin{aligned} \frac{(\rho^- - \rho^+) |V_f^\alpha|_s^{Euler}}{\Delta t} &= (\rho^- - \rho^+) F_f^n \alpha_f^n + (\rho^- - \rho^+) (e_{\alpha_t}(\Delta t) + \frac{1}{\Delta t} e_{\alpha_h}(h^2)), \\ &= (\rho_f^n F_f^n)^{Euler} + (\rho^- - \rho^+) (e_{\alpha_t}(\Delta t) + \frac{1}{\Delta t} e_{\alpha_h}(h^2)). \end{aligned} \quad (8.18)$$

Note that the temporal accuracy lost by dividing by Δt is recovered when the mass flux is integrated over (multiplied with) Δt , from the temporal term in the momentum conservation equation. The next step, therefore, uses the mass flux $(\rho_f^n F_f^n)^{Euler}$ from eq. (8.18) in the discretized convective term from eq. (2.15). Since the unstructured Finite Volume Method linearizes the volumetric flux when discretizing the convective term, an Euler implicit discretization of eq. (2.15) leads to the contribution from the discretized momentum convection term in the form or

$$\frac{\Delta t}{|\Omega_c|} \left(\int_{\Omega_c} \nabla \cdot (\rho \mathbf{v} \mathbf{v}) dV \right)_{t^{n+1}} \approx \frac{\Delta t}{|\Omega_c|} \sum_{f \in F_c} \rho_f^o F_f^o \mathbf{v}_f^{n+1}, \quad (8.19)$$

with the factor $\frac{\Delta t}{|\Omega_c|}$ resulting from the finite-difference approximation of the cell-centered average of the temporal derivative term, i.e. $(\partial_t \rho \mathbf{v})_c$, recovering the first-order temporal accuracy in eq. (8.18) for the implicit Euler temporal discretization scheme physical time T . Note that eq. (2.15) is solved (discretized) iteratively in a segregated solution algorithm (e.g., [2, 47]), so the linearized mass flux is denoted with the outer iteration index $1 \leq o \leq N_o$. If the solution algorithm converges, $\rho_f^o F_f^o = \rho_f^{n+1} F_f^{n+1}$. The Euler implicit scheme in eq. (8.19) allows $\rho_f^o F_f^o$ to be approximated with the consistent mass flux $(\rho_f^o F_f^o)^{Euler}$ from eq. (8.18), which ensures the consistency between mass conservation and volume fraction advection. Namely, in a converged solution, the Euler implicit discretization requires $\rho_f^o F_f^o = \rho_f^{n+1} F_f^{n+1}$, and a single mass flux $(\rho_f F_f)^{Euler}$ can only be obtained by scaling $|V_f^\alpha|_s^{Euler}$ in eq. (8.17), a mass flux averaged over Δt . Therefore, increasing N_o brings the mass flux in eq. (8.19) closer to the one expected by the implicit Euler scheme; however, it does not impact consistency.

The consistency of the mass conservation and volume fraction advection is crucial in the momentum conservation equation, as any deviation from mass conservation in momentum conservation increases the source-term $\sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f [\mathbf{H}(F^o, \rho^o, \mathbf{v}^{i-1})]_f \cdot \mathbf{S}_f$ in the pressure Poisson equation, with i denoting the inner iterations of the pressure equation (cf. [2, 47] for details). In other words, errors in the mass flux artificially accelerate or decelerate the fluid as the pressure equation tries to ensure volume (mass) conservation. Examining the errors in eq. (8.18), it's clear that an error in the mass flux scaled from $|V_f^\alpha|_s$ will be multiplied by $(\rho^- - \rho^+)$: small errors in the volumetric flux (fluxed phase-specific volume) are scaled with the density difference, and this leads to large errors in the velocity field and catastrophic failures for large density differences.

An alternative temporal integration scheme is exemplified for eq. (2.15), say, Crank-Nicolson scheme, resulting in contributions from the convective term in the form of

$$0.5 \frac{\Delta t}{|\Omega_c|} \left(\sum_{f \in F_c} \rho_f^n F_f^n \mathbf{v}_f^n + \sum_{f \in F_c} \rho_f^o F_f^o \mathbf{v}_f^{n+1} \right) \quad (8.20)$$

In this case, it is impossible to ensure consistency. The Crank-Nicolson scheme strictly requires $\rho_f^o F_f^o = \rho_f^{n+1} F_f^{n+1}$, and $(\rho_f F_f)^{Euler}$ is obtained from eq. (8.18) as a single average quantity over Δt , so there are no two mass fluxes for eq. (8.20), making eq. (8.18) inconsistent with the Crank-Nicolson scheme eq. (8.20) already in the first time step.

The main takeaway point is that computing the mass flux by scaling the fluxed phase-specific volume over Δt limits the temporal discretization to schemes that utilize a single mass flux term within Δt - e.g., Euler explicit or implicit, or 2nd-order backward implicit schemes. Any other temporal discretization scheme (e.g., Runge-Kutta) that utilises a linear combination of different mass fluxes within Δt are inconsistent with the mass flux scaling given by eq. (8.18).

Additionally, any modification of the scaled mass flux causes inconsistencies. Concretely, *limiting the mass flux* in the discretized eq. (2.15) causes a hidden inconsistency between mass and volume fraction conservation. The inconsistency is hidden because eq. (8.16) is not solved, a scaled flux is however used, e.g., in eq. (8.18), resulting in the integration of a mass flux (along with momentum) over Δt that is different from the consistent mass flux from eq. (8.18).

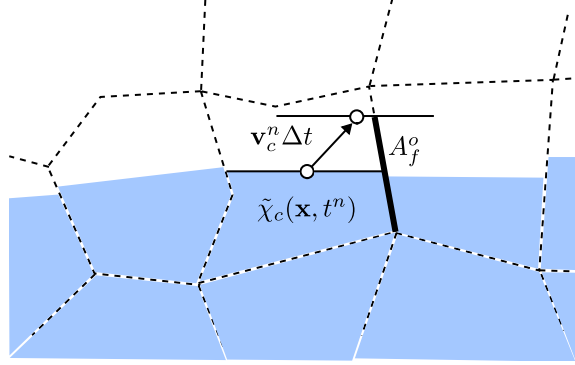


Figure 25: Geometric upwinding for $|V_f^\alpha|_s^{isoAdvect}$ in plicRDF-isoAdvect [6].

In addition to the simplified VOF scheme eq. (8.17), a more complex, geometric isoAdvect scheme [36] is discussed here, that computes

$$\begin{aligned}
 |V_f^\alpha|_s^{isoAdvect} &= \frac{0.5(F_f^n + F_f^o)}{|\mathbf{S}_f|} \int_{t^n}^{t^o} \int_{S_f} \chi(\mathbf{x}, t) dS dt + e_{\alpha_t}(\Delta t^2) + e_{\alpha_h}(h^2) \\
 &= \frac{0.5(F_f^n + F_f^o)}{|\mathbf{S}_f|} \int_{t^n}^{t^o} A_f(t) dt + e_{\alpha_t}(\Delta t^2) + e_{\alpha_h}(h^2) \\
 &= 0.5(F_f^n + F_f^o) \int_{t^n}^{t^o} \alpha_f(t) dt + e_{\alpha_t}(\Delta t^2) + e_{\alpha_h}(h^2)
 \end{aligned} \tag{8.21}$$

with $\int_{t^n}^{t^o} \alpha_f(t) dt$ computed geometrically by displacing the piecewise-linear interface from the upwind cell, using the upwind-cell velocity and the first-order accurate displacement approximation $\mathbf{v}_c \Delta t$, schematically shown in fig. 25. The Euler temporal integration of the displacement $\mathbf{v}_c \Delta t$ for the evaluation of $\int_{t^n}^{t^o} \alpha_f(t) dt$ on the r.h.s. of eq. (8.21) makes it possible to consistently compute the mass flux by scaling the phase-specific fluxed volume

$$(\rho_f^o F_f^o)^{isoAdvect} = \frac{(\rho^- - \rho^+) |V_f^\alpha|_s^{isoAdvect}}{\Delta t}, \tag{8.22}$$

equivalently to eq. (8.18). Even though $\int_{t^n}^{t^o} \alpha_f(t) dt$ is evaluated using exact geometric integration (cf. [6] for details), the A_f^o , used as the end-point of the geometric integration is approximated by first-order displacement $\mathbf{v} \Delta t$, that makes the average displacement velocity

of the fluid interface constant over Δt , and, consequently, the mass flux from eq. (8.22) consistent.

The same consistency would be ensured if V_f^α would be constructed using a flux-based VOF method that maps S_f using the reverse flow-map (e.g., [87]), *if the approximation of the flow-map is first-order accurate*. In other words, if V_f^α is constructed geometrically using displacements given by $\mathbf{v}(\mathbf{x}, t^n)\Delta t$, regardless of the geometrical approximation, V_f^α is constructed using displacements constant over Δt , so dividing the volume V_f^α with Δt results in the consistent volumetric flux, constant over Δt .

Trying to compute the consistent mass flux by scaling the phase-specific fluxed volume for the Crank-Nicolson scheme,

$$\sum_{f \in F_c} 0.5\Delta t(\rho_f^o F_f^o + \rho_f^n F_f^n) = (\rho^- - \rho^+) \sum_{f \in F_c} |V_f^\alpha|_s, \quad (8.23)$$

involves $\rho_f F_f$ at t^n and t^o in the same equation, making it impossible to express either one without the other, using only the known phase-specific volume $|V_f^\alpha|$, fluxed over Δt before the solution of the momentum equation. This confirms the difficulty in evaluating mass flux from time-integrated volumetric fluxes pointed out by Arrufat et al. [76]. Furthermore, in order to integrate $|V_f^\alpha|$ using the Crank-Nicolson scheme (i.e. trapezoidal quadrature), the position of the interface at t^o must be known, namely Σ^o . The interface Σ^o can be reconstructed from α_c^o , which is advected using the forward extrapolation in time (eq. (8.21) and fig. 25, [36]). However, this kind of quadrature does not exactly correspond to the Crank-Nicolson scheme used in the momentum equation, because the advection of the interface Σ^o uses old-time velocities \mathbf{v}_c^n , and Crank-Nicolson in the momentum equation uses the new, implicit velocity \mathbf{v}_c^{n+1} .

If, however, the mass flux is not computed by scaling $|V_f^\alpha|$, $\rho_f F_f$ is uniquely determined by $\chi(\mathbf{x}, t)$ section 3.5.4. Namely, at any time t omitted here for brevity, from eq. (2.10), it exists

$$\int_{S_f} \rho \mathbf{v} \cdot \mathbf{n} dS =: (\rho_f F_f)^\rho \approx \frac{F_f}{|S_f|} \int_{S_f} [(\rho^- - \rho^+) \chi(\mathbf{x}) + \rho^+] dS = (\rho^- - \rho^+) \alpha_f F_f + \rho^+ F_f \quad (8.24)$$

with superscript ρ in $(\rho_f F_f)^\rho$ denoting the mass flux estimated directly from the model for ρ eq. (2.10), i.e. from ρ^\pm and the phase indicator χ (eq. (2.9)). The phase-specific volumetric

flux $\alpha_f F_f$ from eq. (8.24) is, however, inconsistent with the equivalent volumetric flux used for advecting volume fractions eq. (8.12), which can be discretized with a wide range of different flux-based geometric VOF methods. The mass flux $(\rho_f F_f)^\rho$ is different from $(\rho_f F_f)^{isoAdvector}$ eq. (8.22); however, they are both consistent, in the sense that they both satisfy eqs. (8.15) and (8.16). Namely, inserting

$$\rho \mathbf{v} = (\rho^- - \rho^+) \chi \mathbf{v} + \rho^+ \mathbf{v} \quad (8.25)$$

from eq. (2.10) multiplied by \mathbf{v} , into eq. (8.6), and integrating over $[t^n, t^{n+1}]$ results in

$$\rho_c^{n+1} = \rho_c^n - \frac{1}{|\Omega_c|} \int_{t^n}^{t^{n+1}} \left[\int_{\partial\Omega_c} (\rho^- - \rho^+) \chi \mathbf{v} \cdot \mathbf{n} dS dt + \rho^+ \int_{\partial\Omega_c} \mathbf{v} \cdot \mathbf{n} dS \right] dt \quad (8.26)$$

$$= \rho_c^n - \frac{1}{|\Omega_c|} \int_{t^n}^{t^{n+1}} \left[\int_{\partial\Omega_c} (\rho^- - \rho^+) \chi \mathbf{v} \cdot \mathbf{n} dS + \rho^+ \int_{\Omega_c} (\nabla \cdot \mathbf{v}) dV \right] dt \quad (8.27)$$

$$= \rho_c^n - \frac{1}{|\Omega_c|} \int_{t^n}^{t^{n+1}} \left[\int_{\partial\Omega_c} (\rho^- - \rho^+) \chi \mathbf{v} \cdot \mathbf{n} dS \right] dt \quad (8.28)$$

$$= \rho_c^n - \frac{(\rho^- - \rho^+)}{|\Omega_c|} \sum_{f \in F_c} \int_{t^n}^{t^{n+1}} \chi \mathbf{v} \cdot \mathbf{n} dS \quad (8.29)$$

$$= \rho_c^n - \frac{(\rho^- - \rho^+)}{|\Omega_c|} \sum_{f \in F_c} |V_f^\alpha|_s. \quad (8.30)$$

The main point of eq. (8.30) is that the additional term $\rho^+ \mathbf{v}$, discrete $\rho^+ F_f$ in eq. (8.24), does not impact consistency requirement of eqs. (8.15) and (8.16) - ensuring that the mass conservation remains equivalent to volume (phase indicator, volume fraction) conservation, scaled with the density difference. More importantly, eq. (8.30) shows this is true irrespective of the VOF method used to approximate $|V_f^\alpha|_s$, in our case, the plicRDF-isoAdvector method [6]. The second consistency requirement to use exactly the same $\rho_f F_f$ in the discrete momentum equation is still necessary.

8.2 Collocated segregated solution algorithm with the auxiliary density equation

The consistent solution algorithms are adapted for the unstructured Volume-of-Fluid methods on the plicRDF-isoAdvector method [6], and the auxiliary density equation solution is imple-

mented into the segregated algorithm (i.e. solver) "interIsoRhoFoam", which is summarized by algorithm 6.

In section 3.5.4, it is shown on the level of the mathematical model why solving an auxiliary mass conservation equation plays a key role in reducing numerical inconsistency caused by high density ratios.

The mass conservation equation is solved in the outer loop of the segregated algorithm algorithm 6 in the following discrete form

$$\rho_c^{o+1} = \rho_c^o + \frac{\Delta t}{|V_{\Omega_c}|} \sum_f \rho_f^o F_f^o. \quad (8.31)$$

Equation (8.31) is the auxiliary mass conservation (density equation). It is solved after updating the volume fraction in the outer loop to α_c^o by utilizing eq. (8.12) with any flux-based VOF method. Interface reconstruction computes $\tilde{\chi}_c(\mathbf{x}, t^o)$ in every finite volume Ω_c intersected by the fluid interface. The piecewise-linear interface approximation $\tilde{\chi}_c(\mathbf{x}, t^o)$, together with eq. (2.10), provides ρ_f^o for eq. (8.31). Since F_f^o is also available, eq. (8.31) can be explicitly evaluated. To compute the mass flux, the consistency relationship between the phase indicator and the face-centered density ρ_f is utilized, derived in section 3.5.4 for the hybrid Level Set / Front Tracking method. It can be applied on any two-phase flow simulation method that is using a phase indicator χ . The surface integral of mass flux at time step o is expressed as

$$\int_{\partial\Omega_c} \rho^o \mathbf{v}^o \cdot \mathbf{n} dS = \int_{\partial\Omega_c} [\rho^- \chi^o + \rho^+ (1 - \chi^o)] \mathbf{v}^o \cdot \mathbf{n} dS, \quad (8.32)$$

and discretized as

$$\sum_{f \in F_c} \rho_f^o F_f^o = \sum_{f \in F_c} [\rho^- \alpha_f^o + \rho^+ (1 - \alpha_f^o)] F_f^o, \quad (8.33)$$

where

$$\alpha_f^o = \frac{1}{|S_f|} \int_{S_f} \chi(\mathbf{x}, t^o) dS = \frac{|A_f(t^o)|}{|S_f|}. \quad (8.34)$$

is used to define the face-centered density

$$\rho_f^o = \rho^- \alpha_f^o + \rho^+ (1 - \alpha_f^o) \quad (8.35)$$

in the mass flux. The momentum equation is discretized as

$$\rho_c^o \mathbf{v}^o - \rho_c^n \mathbf{v}^n + \frac{\Delta t}{|\Omega_c|} \sum_f \rho_f^o F_f^o \mathbf{v}_f^o = \frac{\Delta t}{|\Omega_c|} \mathbf{M}, \quad (8.36)$$

and solved for \mathbf{v}_f^o in the outer loop of the $p - \mathbf{v}$ solution algorithm. The source term \mathbf{M} is a shorthand term that contains all the remaining contributions from the discretization, used here for brevity. The volume fractions α_c^o are already solved for using eq. (8.12) at the beginning of the "o" outer iteration, and used to approximate the phase indicator $\tilde{\chi}_c(\mathbf{x}, t^o)$ for the mass flux $\rho_f^o F_f^o$ using eq. (8.33), used in the same way in the auxiliary density equation eq. (8.31), solved for ρ_c^o , as $\rho_f^o F_f^o$ in eq. (8.36), *without using flux limiters*. The values computed from the last outer loop are regarded as the new values at the time step. At last, the density field is needed to be restored with respect to the volume fraction to maintain consistency between them. The solution algorithm `interIsoRhoFoam` is summarized by algorithm 6. Algorithm 6 uses a combination of SIMPLE and PISO algorithms in OpenFOAM[®] with a residual-based control to terminate outer iterations, which tests if a maximal number of iterations has been reached, the final domain-maximal residuals of the pressure equation r_p are below absolute tolerance, or the ratio of the final and initial domain-maximal residuals $\frac{r_p}{r_p^i}$ is smaller than a user-prescribed relative tolerance.

Algorithm 6 The solution algorithm `interIsoRhoFoam`.

```

1: while  $t \leq t_{end}$  do
2:    $t^{n+1} = t^n + \Delta t$ 
3:   for  $o = 0; o < N_{outer}; ++o$  do
4:     Solve volume fraction for  $\alpha_c^o$  ▷ Equation (8.12)
5:     Reconstruct the phase indicator  $\tilde{\chi}_c(\mathbf{x}, t^o)$ 
6:     Compute  $\rho_f^o$  from  $\tilde{\chi}_f(\mathbf{x}, t^o)$  ▷ Equation (8.35)
7:     Compute the mass flux  $\rho_f^o F_f^o := \rho_f^o (\mathbf{v}_f^{o-1} \cdot \mathbf{S}_f)$ 
8:     Solve the density for  $\rho_c^o$  with  $\rho_f^o F_f^o$  ▷ Equation (8.31)
9:     Discretize momentum equation (Equation (2.15)) with  $\rho_c^o$  and  $\rho_f^o F_f^o$ .
10:    for  $i = 0; i < N_{inner}; ++i$  do
11:      Solve the pressure equation for  $p_c^i$ . ▷ Cf. [2, 47].
12:      Compute  $F_f^i$  and  $\mathbf{v}_c^i$  from  $p_c^i$ . ▷ Cf. [2, 47].
13:    end for
14:  end for
15:  Restore  $\rho_c^{n+1}$  consistent with  $\alpha_c^{n+1}$ , i.e.  $\rho_c^{n+1} = (\rho^- - \rho^+) \alpha_c^{n+1} + \rho^+$  ▷ Equation (2.10).
16: end while

```

8.3 Phase-specific face area calculation

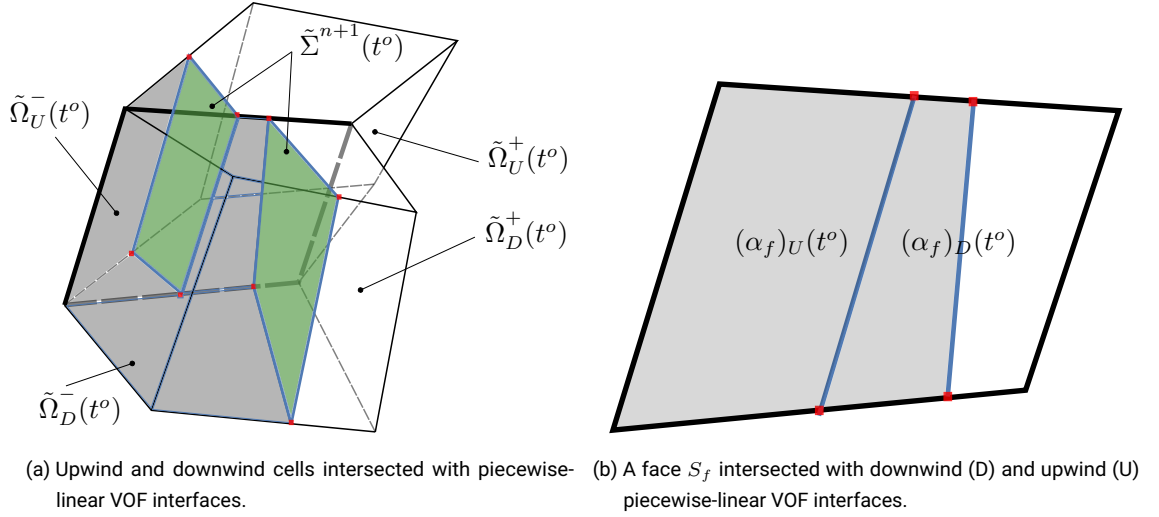


Figure 26: Interface reconstructed as $\tilde{\Sigma}_{U,D}^o$ in upwind (U) and downwind (D) cells. Green polygons are interface polygons $\tilde{\Sigma}_{U,D}^o \cap \Omega_{U,D}$. Blue lines are intersection line segments $\tilde{\Sigma}_{U,D}^o \cap S_f$. Red points are intersection points $\tilde{\Sigma}_{U,D}^o \cap \partial S_f$.

The integral eq. (8.34) leaves room for alternative discretizations, and therefore requires attention. Flux-based geometrical Volume-of-Fluid methods advect volume fractions using geometrical upwind advection schemes, transporting $\tilde{\Omega}^-(t) \approx \Omega^-(t)$ geometrically from the upwind cell, denoted with U in fig. 26a, to the downwind cell, denoted with D in fig. 26a. Flux-based geometrical VOF methods therefore already provide the intersection points and the intersection line segments denoted as red dots and the blue line segments in fig. 26, are available, simplifying the area fraction calculation in eq. (8.34). Since the geometrical VOF methods approximate the phase indicator as piece-wise continuous, with a jump discontinuity not only across the fluid interface $\Sigma(t)$ but also across the finite volume boundary $\partial\Omega_c$, the line segments forming the intersection of the piece-wise continuous interface and a face S_f it intersects (see fig. 26a), do not overlap. fig. 26b shows schematically the two intersection line segments on a cell face. The intersection line segment of the interface and the upwind

cell are selected as the submerged area border to calculate the area fraction. The evaluation of α_f and, from it, ρ_f, μ_f , is summarized by algorithm 7.

Algorithm 7 Sub-algorithm for calculating α_f, ρ_f, μ_f .

- 1: Initialize ρ_f, α_f with upwind cell-centered values ρ_U, α_U .
 - 2: $\alpha_f = \frac{1}{|S_f|} \int_{S_f} \tilde{\chi}_U(\mathbf{x}) dS = \frac{|\tilde{\Omega}_U^- \cap S_f|}{|S_f|}$ ▷ Equation (8.34) with upwind χ .
 - 3: $\rho_f = (\rho^- - \rho^+) \alpha_f + \rho^+$. ▷ Equation (2.10).
 - 4: $\mu_f = (\rho^- \nu^- - \rho^+ \nu^+) \alpha_f + \rho^+ \nu^+$. ▷ Equation (2.11).
-

It is important to note that μ_f is geometrically evaluated from a geometrical α_f , and not interpolated, as it is found that interpolation leads to large errors in simulations with large differences in dynamic viscosity.

8.4 Consistency of VOF methods for Two-Phase Flows with High Density Ratios

Here the findings that lead to an equation discretization with flux-based geometrical VOF methods that remains consistent for very high density ratios are summarized:

- Computing the mass flux $\rho_f F_f$ by scaling the fluxed phase-specific volume $|V_f^\alpha|_s$ with $(\rho^- - \rho^+)/\Delta t$ approximates only one constant average mass flux over Δt .
- The constant mass flux $\rho_f F_f$ scaled from $|V_f^\alpha|_s$ with Δt , as an average value over Δt , can only be consistently used in first-order schemes, i.e. use a single mass flux value over Δt .
- Computing the mass flux $\rho_f F_f$ from the density model eq. (2.10) disconnects mass conservation from volume fraction advection, which uses geometrical integration, i.e. a geometrically integrated phase-specific volumetric flux. This requires a solution of an additional (auxiliary) density equation ([47, 51]) for cell-centered density, discarded at the end of the time step.

- Upwinding geometric VOF methods that use the Euler temporal scheme to approximate point displacements are equivalent to using $\rho_f F_f$ from eq. (8.24) and solving the density equation for ρ_c^o , used further in $p - \mathbf{v}$ coupling to obtain a divergence-free cell-centered velocity. In other words, combining Euler temporal integration scheme with upwind scheme for the momentum equation guarantees numerical consistency for any flux-based VOF method, which uses temporally first-order accurate displacements in its geometrical integration of the fluxed phase-specific volume.

These points are verified and validated in the following section.

9 Verification and validation

Data archives of the implementation of the interIsoRhoFoam algorithm, input data, post-processing software and secondary data are publicly available [123, 124]. The method is actively developed in a publicly available git repository [125].

9.1 Time step size

The time step is limited by the CFL condition in the explicit plicRDF-isoAdvector method [6],

$$\Delta t_{CFL} = \frac{CFL h}{|\mathbf{v}|}, \quad (9.1)$$

where h is the discretization length, and $CFL = 0.2$ is used from [6]. Another restriction for the time step size considers the propagation of capillary waves on fluid interfaces,

$$\Delta t_{cw} = \sqrt{\frac{(\rho^+ + \rho^-)h^3}{2\pi\sigma}}. \quad (9.2)$$

This time step constraint was introduced first by Brackbill, Kothe, and Zemach [27], and revised by Denner and Wachem [97]. In this study, the time step is restricted using the relation from Tolle, Bothe, and Marić [2], namely

$$\Delta t = \min(k_1 \Delta t_{cw}, k_2 \Delta t_{CFL}) \quad (9.3)$$

where k_1 and k_2 are scaling factors. $k_1 = 1$, $k_2 = 0.5$ are give as the default value in this section.

9.2 Translating droplet in ambient flow

A canonical test case, originally introduced by Bussmann, Kothe, and Sicilian [58] in 2D, involves a moving droplet in quiescent ambient flow. Zuzio et al. [7] extended the 2D case to 3D using a density ratio of 10^6 . Following the setup in [7], the droplet of radius $R = 0.15$ has the initial velocity of $(0, 0, 10)$. To smooth the velocity field and avoid the perturbation caused by sudden acceleration of still ambient flow, the initial constant velocity is assigned not only for the cells of the droplet and the interface layer but also the interface cell layer adjacent to interface cells. The droplet with initial center location $(0.5, 0.5, 0.5)$ translates to a distance of $L = 1$ and the simulation time is then $t_{end} = 0.1$. The computational domain has dimensions $L_z = L_x = L_y = 1$. The periodic boundary condition is applied to all boundary patches. The tests with the periodic boundary condition showed that the plicRDF-isoAdvector method [6] implemented in [49] has an inconsistency at the periodic boundary, which is fixed as described in appendix 3. The mesh setup from [7] is also followed: the mesh resolution is in a range of $N \in (32, 48, 64)$ per unit side-length of the computational domain, resulting in $\approx (10, 15, 20)$ mesh cells per droplet diameter. Surface tension force is neglected in this case, so only Δt_{CFL} from eq. (9.3) is taken into account. The viscosity and gravitational forces are neglected to highlight the numerically consistent behavior of the mass and momentum convection. Three error norms are adopted to evaluate the results quantitatively, for mass, momentum, and sphericity:

$$E_{mass} = \frac{M(t) - M(0)}{M(0)} = \frac{\sum_k m_k(t) - \sum_k m_k(0)}{\sum_k m_k(0)} = \frac{\sum_k \rho_k(t)V_k}{\sum_k \rho_k(0)V_k}, \quad (9.4)$$

$$E_{mom} = \frac{|\sum_k m_k(t)\mathbf{v}_k(t)| - |\sum_k m_k(0)\mathbf{v}_k(0)|}{|\sum_k m_k(0)\mathbf{v}_k(0)|}, \quad (9.5)$$

$$E_{sph} = \left| \sum_{c \in C} S_c(t) - \sum_{c \in C} S_c(0) \right|, \quad (9.6)$$

where the subscript k indicates that the value is extracted from the cell Ω_k and the m_k, V_k denote the mass and volume of Ω_k . In eq. (9.6), S_c is the area of the PLIC-VOF interface polygon in the cell Ω_c . These two ratios of eqs. (9.4) and (9.5) represent the time evolution of the normalized error of the global sum of the heavy phase mass and momentum. In this case, E_{mass} is expected to be near zero because there is no source and dissipation for both mass, and flux-based geometrical VOF method have a very high degree of local volume (mass) conservation.

In the absence of force terms on both sides of momentum transport equation, i.e., eq. (2.15), assuming the periodic boundary condition is applied to all boundary patches, momentum conservation dictates that the deviation E_{mom} should also theoretically remain at zero over time. Bussmann, Kothe, and Sicilian [58] proposed that a droplet, characterized by a large density ratio (10^6), should undergo translation without deformation in an ambient flow, much like a solid sphere moving through a void. This conclusion has been widely accepted and corroborated by several publications (e.g., [50, 51, 53, 55, 65]). However, these studies qualitatively assessed droplet deformation based on visual representations of droplet shape. In this work, the sphericity Ψ_d is employed as a quantitative measure. Additionally, E_{sph} is used to characterize the deviation of deformation from the initial droplet shape.

As discussed in section 8.4, when the first-order accurate Euler and Gauss upwind scheme are employed to discretize momentum conservation eq. (2.15), the mass flux $(\rho_f F_f)^{isoAdvect}$ from eq. (8.22) will be consistent. Since the choice of discretization schemes ensures consistency of the discretization, there is no need to modify the implementation of the numerical method. The analysis from section 8 has been verified for the "interIsoFoam" solver and it has been compared with the "interIsoRhoFoam" solution algorithm 6 that implements the auxiliary density equation. The normalized mass error eq. (9.4) is nearing machine epsilon, and is therefore much smaller than the linear solver tolerance (set for this case to 10^{-12}), for both interIsoFoam and interIsoRhoFoam, showing excellent conservation of mass for both configurations. The consistency of the mass and phase indicator transport, as well as the consistency of the mass flux approximation of the interIsoFoam and interIsoRhoFoam with Euler+upwind schemes is reflected in an equivalent accuracy and stability for the momentum: eq. (9.5) remains equally much smaller than the linear solver tolerance, nearing machine epsilon, and remains stable. Therefore, the consistency and equivalence of the "interIsoFoam"

Time schemes	Gaussian convection schemes	Order of convection accuracy	Category of convection	Boundedness of convection	Mass flux consistency
Crank-Nicolson[126]	upwind	first-order	NVD/TVD	Bounded	no
	upwind	first-order	NVD/TVD	Bounded	yes
Euler	cubic[127]	second-order	non-NVD/TVD	Unbounded	no
	limitedLinearV	first-/second-order	NVD/TVD	Unbounded	no
	linear	second-order	non-NVD/TVD	Unbounded	no
	LUST	second-order	non-NVD/TVD	Unbounded	no
	MUSCL[128]	second-order	NVD/TVD	Unbounded	no
	QUICK[129]	second-order	NVD/TVD	Unbounded	no
	SuperBee[130]	second-order	NVD/TVD	Unbounded	no
vanLeer[131]	second-order	NVD/TVD	Unbounded	no	

Table 5: The combinations of different time and convection schemes used to test the effect of numerical consistency.

and "interIsoRhoFoam" are verified, with the Euler+upwind schemes.

Note that this verification case is extremely challenging, since it is an inviscid case - there is no viscous force available in this case to dampen the errors resulting from inconsistent two-phase mass and momentum transport.

Next, inconsistencies leading to large errors and often to catastrophic failure when more than one mass flux is used over Δt when discretizing eq. (2.15) are demonstrated, or the mass flux is limited in the discretized eq. (2.15).

9.2.1 Comparison of different schemes

Using the Crank-Nicolson scheme to discretize eq. (2.15) reveals that the temporal scheme involving implicit mass flux $\rho_f F_f$ results in a mismatch between mass convection and volume fraction convection scaled by $(\rho^- - \rho^+)$, due to the fact that the Navier - Stokes equation using a segregated method is solved iteratively within a time step, and the interface's advection velocity thus cannot be updated simultaneously with \mathbf{v}^{n+1} from the previous $p - v$ coupling iteration. The inconsistency is amplified by the density-ratio. Combinations of schemes listed in table 5, are tested to verify their effect on the mass flux inconsistency.

Figures 27 and 28 represent the temporal evolution of E_{mass} , and E_{mom} . It is evident from fig. 27 that mass conservation is not maintained when utilizing the Crank-Nicolson time discretization scheme, and divergence schemes cubic, Linear-Upwind Stabilised Transport

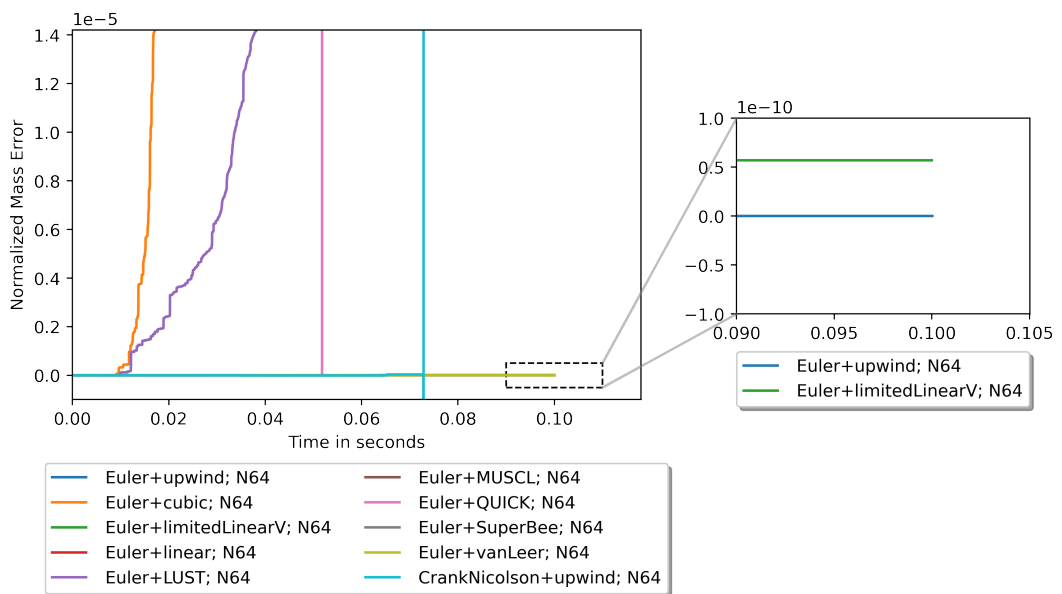


Figure 27: Temporal evolution of normalized mass conservation error with different schemes: interIsoFoam, $N = 64$.

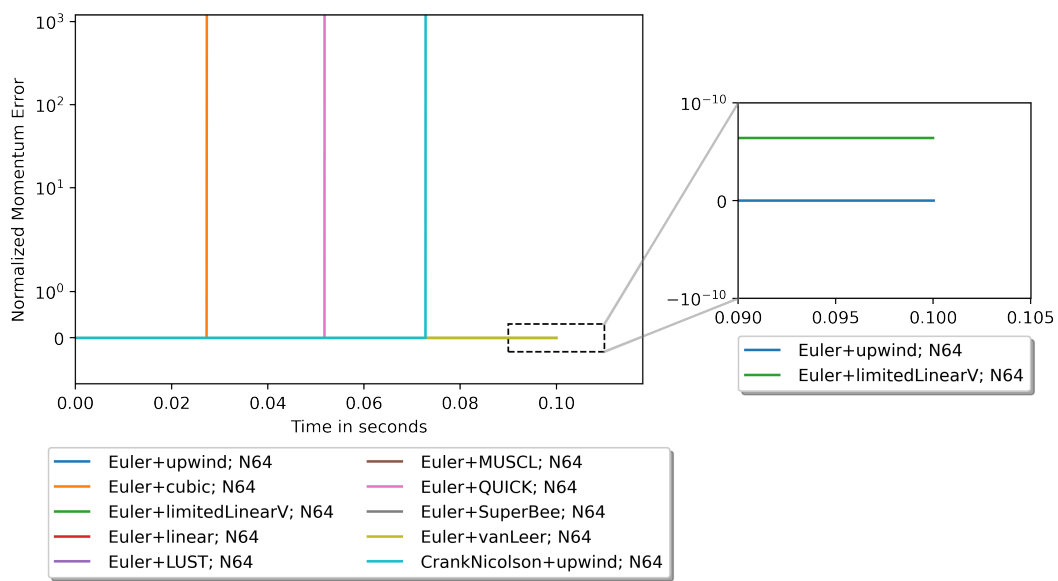


Figure 28: Temporal evolution of normalized momentum conservation error with different schemes: interIsoFoam, $N = 64$.

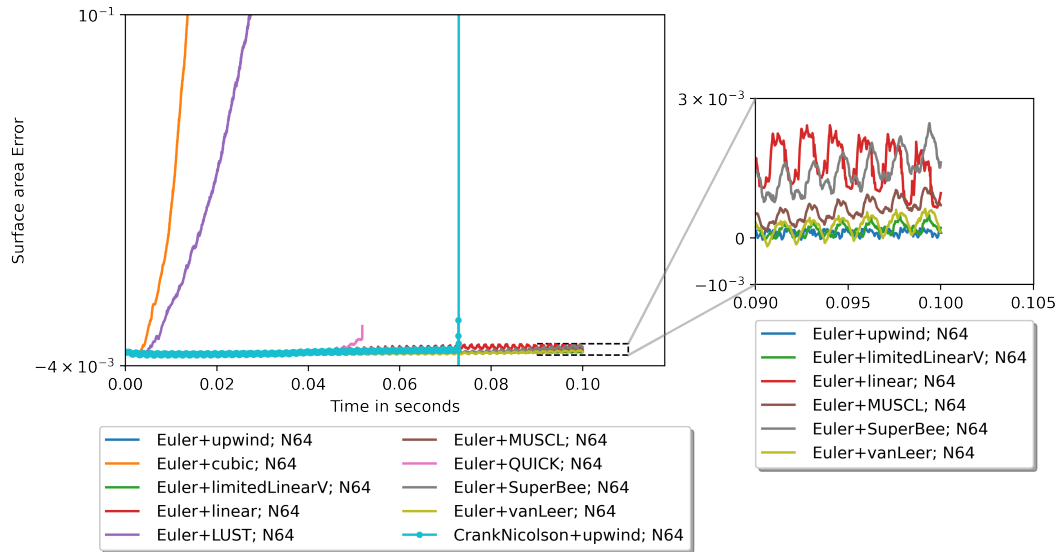
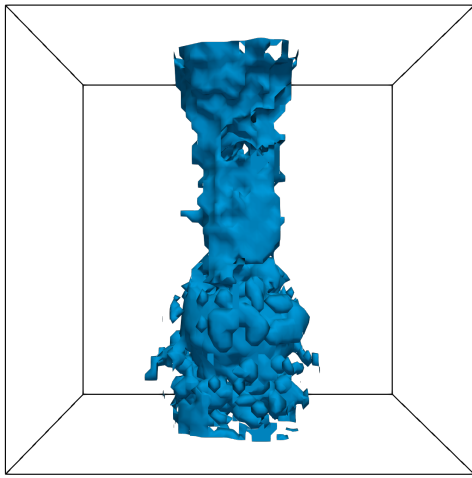


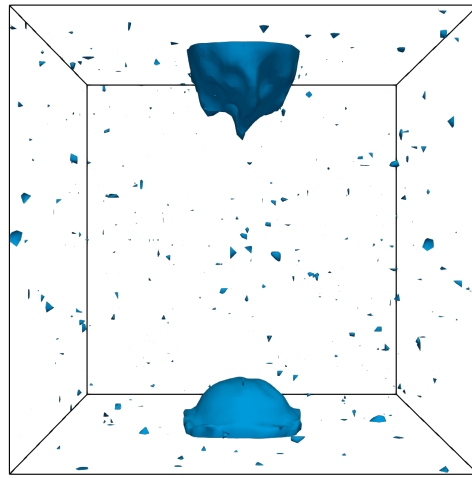
Figure 29: Temporal evolution of sphericity error with different schemes: interIsoFoam, $N = 64$.

(LUST) and Quadratic Upstream Interpolation for Convective Kinematics (QUICK). These simulations terminate at an early stage with catastrophic failure. However, for cases that can run until the final time, the magnitude of mass errors, as shown in the zoomed subfigure of fig. 27, is on the order of 10^{-10} , indicating mass conservation. This observation aligns with the inherent characteristic of the volume of fluid method, which is known for its mass conservation property. The vertical lines from the results of cubic, QUICK and Crank-Nicolson can be observed in fig. 28. Some combinations deliver stable momentum errors. A closer examination of the stable cases in the zoomed view of fig. 28 confirms the use of the Euler temporal scheme.

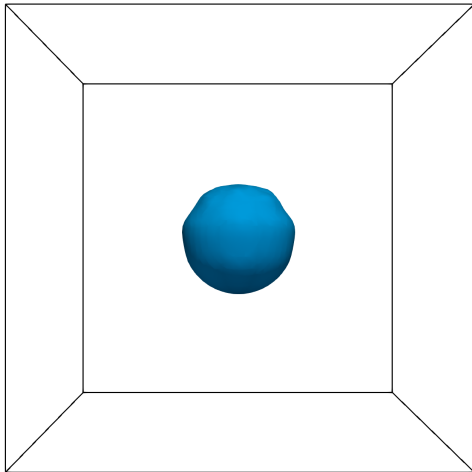
The temporal evolution of sphericity error is depicted in fig. 29 where the steepness of the curves indicates the extent of droplet deformation from its initial shape. The results obtained from four unstable scheme combinations show significantly larger deviations in sphericity. It is notable that although the schemes Euler+limitedLinearV, linear and vanLeer, demonstrate comparable performances compared with the consistent-on-paper Euler+upwind in terms of



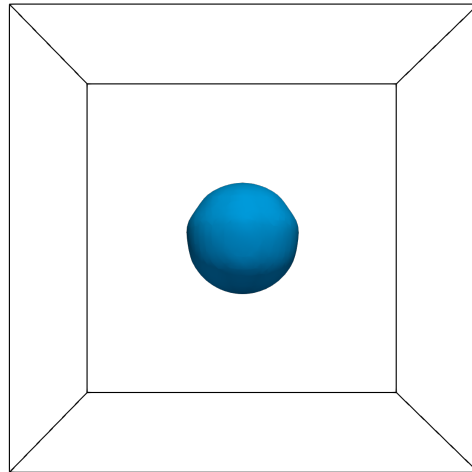
(a) Crank-Nicolson + upwind



(b) Euler + LUST



(c) Euler + limitLinearV



(d) Euler + upwind

Figure 30: Final shape of the droplet calculated by interIsoFoam with different schemes:
 $N = 64$.

mass and momentum conservation in figs. 27 and 28, sphericity errors calculated using these schemes deviate more from the expected value. Moreover, the zoomed view reveals that,

apart from Euler+upwind, all other stable scheme combinations display varying degrees of divergence. This suggests that if the simulations were to continue for a longer duration, these cases would likely crash. Figure 30 presents the final shapes of the droplets simulated using interIsoFoam with two stable scheme combinations and two unstable combinations. For the unstable combinations, a common occurrence is observed: the original droplets disintegrate into small, irregularly scattered pieces. A comparison between the final shapes obtained using the inconsistent but in this verification case, for these parameters, still stable scheme combination Euler + limitedLinearV (fig. 30c) and the consistent scheme combination Euler + upwind (fig. 30d) corroborates the findings from the sphericity errors in fig. 29: an evident "crown" with irregular bumps forms on the top part of the droplet in fig. 30c, leading to a reduction in sphericity. On the contrary, in fig. 30d, only a slight shrinkage occurs in the neck region of the droplet, while the top surface remains smooth.

table 6 summarizes these findings, providing information on early termination of scheme combinations that are inconsistent, as well as mass, momentum and sphericity errors for different mesh resolutions. Two finer resolutions, i.e., $N \in (96, 128)$, are tested for interIsoFoam with Euler+upwind schemes to verify the mesh convergence of the consistent method. The study shows the clear tendency of error reducing with the finer mesh, as illustrated in fig. 31.

All scheme combinations are tested for a low density ratio with interIsoFoam, i.e., density ratio = 1. As shown in fig. 51, all cases run stably to the terminated time, and the conservation errors are reduced to below the machine epsilon.

9.3 Translating sub-millimeter droplet with realistic physical properties

materials/properties (25 °C)	density (kg/m ³)	kinematic viscosity (m ² /s)	surface tension (N/m)	density ratio
air	1.1839	---	---	--- [132]
mercury	13.5336×10^3	---	--- (in air)	11431.37(in air) [132]

Table 7: Realistic fluid properties of the mercury droplet/air ambient pair.

The realistic densities of the mercury droplet and air ambient pair are selected for the verification case of the translating sub-millimeter droplet. The density ratio displayed in table 2

Scheme combination	Resolution	interIsoRhoFoam				interIsoFoam			
		End time	Mass error	Momentum error	Sphericity Error	End time	Mass error	Momentum error	Sphericity Error
Euler+upwind	32	0.1	0.0	0.0	3.062e-03	0.1	0.0	0.0	3.062e-03
	48	0.1	0.0	0.0	1.111e-03	0.1	0.0	0.0	1.110e-03
	64	0.1	0.0	0.0	3.484e-04	0.1	0.0	0.0	3.484e-04
	96	*	*	*	*	0.1	0.0	0.0	3.197e-04
	128	*	*	*	*	0.1	0.0	0.0	2.298e-04
Euler+cubic	32	0.0206	4.840	1.028e+18	-0.1299	0.067	8.807e-03	1.440e+10	0.3501
	48	0.01	-2.925e-02	3.581e+22	-0.2341	0.0260	1.6346	2.390e+6	0.1024
	64	0.0065	-2.190e-03	2.233e+22	-0.2267	0.0273	0.8301	3.142e+8	0.6643
Euler+limitedLinearV	32	0.1	0.0	5.511e-04	6.773e-03	0.1	0.0	-2.656e-09	5882e-3
	48	0.1	0.0	4.793e-04	2.852e-03	0.1	0.0	-4.995e-09	3.671e-03
	64	0.1	0.0	3.290e-04	2.343e-03	0.1	5.695e-11	6.407e-11	2.216e-04
Euler+linear	32	0.0384	-4.520e-04	3.603e+21	-0.1932	0.1	0.0	-2.540e-10	2.234e-02
	48	0.0306	-0.1954	4.271e+24	-0.2330	0.1	0.0	2.210e-07	1.843e-02
	64	0.0239	-0.0676	2.604e+25	-0.2481	0.1	1.210e-09	3.915e-10	9.681e-4
Euler+LUST	32	0.0221	598.0	3.036e+21	-0.2055	0.0993	0.1097	2.162e+19	0.4242
	48	0.0166	1.026e+55	3.272e+120	0.2428	0.0258	1.047e+13	1.797e+39	-0.1762
	64	0.002	0.0	9.254e-05	5.042e-04	0.0485	2.006e-05	-2.664e-05	0.2856
Euler+MUSCL	32	0.0046	0.0	4.727e+8	-0.0932	0.1	0.0	1.143e-05	7.383e-03
	48	0.0064	0.0	-0.1949	-3.500e-03	0.0485	2.227e+12	2.566e+36	-0.1434
	64	0.0092	0.0	1.059e-03	-1.202e-03	0.1	3.075e-09	1.780e-07	7.070e-4
Euler+QUICK	32	0.0203	0.0	5.455e+6	-0.2096	0.1	0.0	5.177e-4	9.6670e-03
	48	0.0066	-3.198e-09	6.806e+82	-0.2699	0.0185	2.417e-06	4.164e+51	0.0118
	64	0.0079	-5.097e-07	1.862e+24	-0.1086	0.0518	0.1564	5.537e+21	7.812e-3
Euler+SuperBee	32	0.0237	0.0	5.732e+15	-0.1570	0.1	0.0	-2.632e-05	5.094e-03
	48	0.0022	0.0	0.2876	-4.630e-04	0.0631	1.645e-10	8.588e+42	0.0120
	64	0.0075	0.0	-1.206e-04	-8.018e-04	0.1	1.477e-08	2.970e-07	1.628e-03
Euler+vanLeer	32	0.1	0.0	4.190e-04	-4.514e-04	0.1	0.0	4.916e-07	3.495e-03
	48	0.1	0.0	3.480e-04	1.417e-03	0.1	0.0	3.535e-07	4.868e-03
	64	0.0896	0.0	-9.273e-05	0.0107	0.1	3.310e-09	5.478e-08	1.388e-04
CrankNicolson+upwind	32	0.0096	0.0	4.131e-04	-9.018e-04	0.0043	-0.2219	7.627e+21	-0.1632
	48	0.0049	0.0	3.238e-03	-3.139e-04	0.0037	32.30	2.203e+24	-0.1965
	64	0.0032	6.001e-03	6.280e+04	-2.093e-3	0.0729	9.124e+25	2.576e+60	0.3248

Table 6: The terminated time and the final normalized errors of interIsoRhoFoam and interIsoFoam with different scheme combinations and mesh resolutions for translating droplet in ambient flow case.

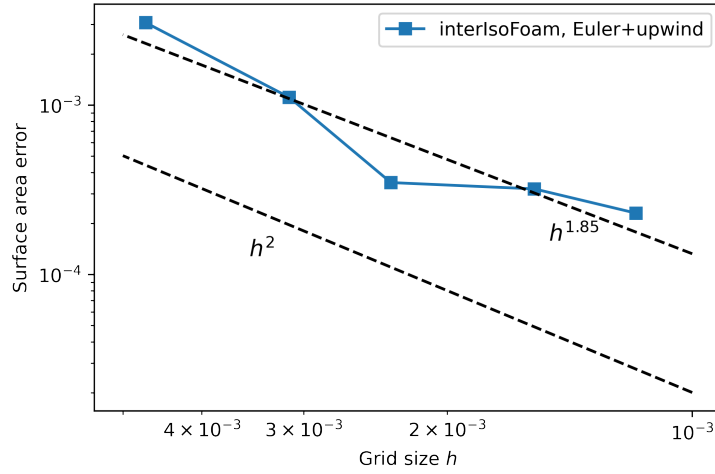


Figure 31: Mesh convergence study for sphericity error: interIsoFoam, Euler+upwind, $N \in (36, 48, 64, 96, 128)$.

is around 10^4 . The rest setups are same as in [47]. A spherical droplet of radius $R = 0.25$ mm translates a distance of three diameters with velocity 0.01 m/s in z -direction of the rectangular solution domain ($L_x = L_y = 5R, L_z = 15R$). The ambient flow has the same velocity, that is $\mathbf{v}_a = (0, 0, 0.01)$. Three resolutions are tested in this case: $N \in (16, 32, 64, 96, 128)$. The initial centroid position of the droplet is $(2.5R, 2.5R, 2R)$. Surface tension and viscous forces are not considered in this case. Since the droplet translates with the ambient flow and there is no sink or source for the droplet moving, the velocity field should keep unchanged.

The error norm L_∞ is employed to measure the maximal deviation between the numerical velocity and the analytical one among all cells, i.e.,

$$L_\infty(\mathbf{v}) = \max_i \left(\frac{\|\mathbf{v}_i - \mathbf{v}_\infty\|}{\|\mathbf{v}_\infty\|} \right), \quad (9.7)$$

where \mathbf{v}_i denotes the velocity of the cell i , and the analytical velocity value is $\mathbf{v}_\infty = \mathbf{v}_a = (0, 0, 0.01)$. The expected exact value of L_∞ is 0; however, in practice, the absolute accuracy is limited by the absolute tolerance of the linear solver used to solve the pressure Poisson equation.

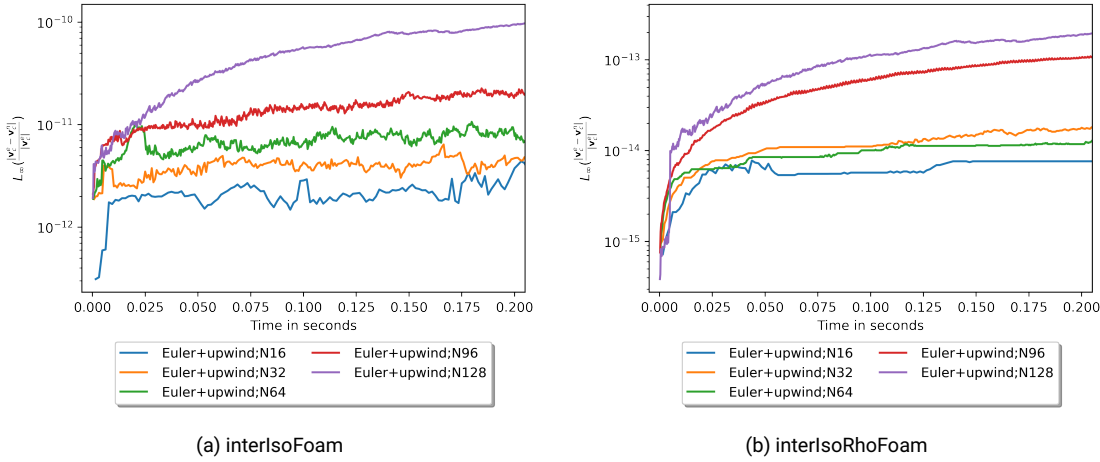


Figure 32: Temporal evolution of the velocity error norm $L_\infty(\mathbf{v})$ with pure advection: Euler, Gauss upwind, density ratio: 10^4 , mesh resolution: $N \in (16, 32, 64, 96, 128)$.

Figure 32 presents the temporal evolution of $L_\infty(\mathbf{v})$. The same $L_\infty(\mathbf{v})$ calculated for both solvers reveals a very close numerical equivalence between the volume fraction and mass conservation equation using the Euler+upwind combination of schemes. Errors of both solvers remain stable. Absolute errors of interIsoFoam are somewhat larger; however, they remain in the realm of numerical noise, significantly below the linear tolerance for the pressure Poisson equation, ensuring consistency. A notable outcome from fig. 32 is the value of the final converged $L_\infty(\mathbf{v})$, which is at the magnitude of 1×10^{-11} and for interIsoRhoFoam almost reaches the machine epsilon, confirming numerical stability and consistency of a very high degree for this challenging verification case.

Additional tests are conducted on all schemes listed in table 5 for this particular case setup. The corresponding results are presented in fig. 33. The results from Crank-Nicolson temporal scheme and SuperBee convection scheme show significant numerical instability at the initial stage of the simulation. In contrast, the errors obtained from all other schemes remain stable throughout the entire simulation. Notably, there is a substantial variation in accuracy among these schemes. The velocity errors from Euler + QUICK and upwind remain at magnitudes around 10^{-11} , while the errors from the other schemes initially increase and

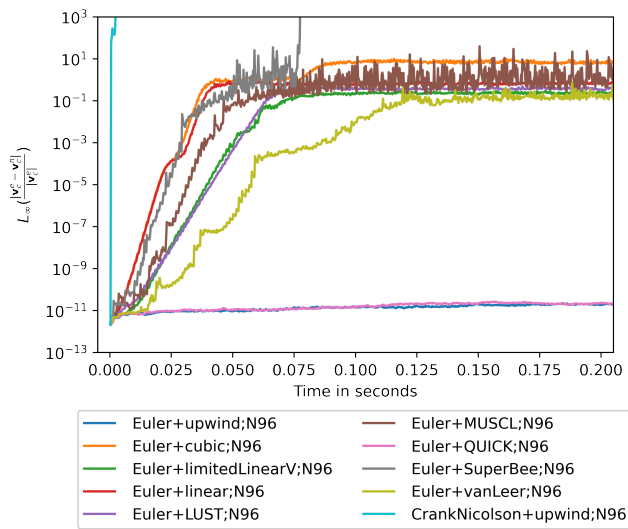


Figure 33: Temporal evolution of the velocity error norm $L_\infty(\mathbf{v})$ with pure advection - combining 10 schemes, density ratio is 10^4 , mesh resolution is $N = 96$. Only Euler and upwind(ing) schemes remain consistent and stable.

stabilize at values that are 10^7 to 10^{10} times larger. Additionally, the same cases are tested with density ratio 1 to investigate the effects of the density ratio on the numerical instability. As shown in fig. 52, all cases keep stable with the low density ratio at the final stage. The velocity errors of all tests are reduced to magnitudes of 10^{-14} , even for the most critical schemes, i.e. Crank-Nicolson + upwind.

9.4 Mixing layer

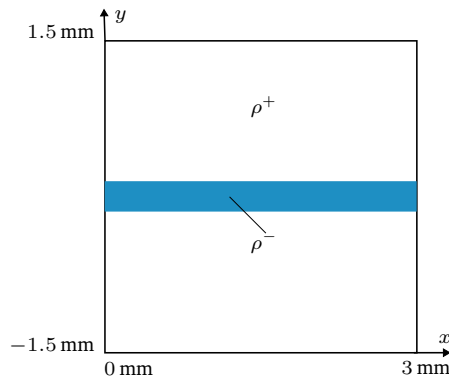


Figure 34: 2D mixing layer

In this case, a 2D mixing layer case is tested. The 2D computational domain as depicted in fig. 34 has the same length $L = 3$ mm in both x - and y -direction. The liquid with high density $\rho^- = 1000$ kg/m³ flows in the middle region -0.15 mm $< y < 0.15$ mm of the square computational domain with relatively low initial velocity $v_x^- = 2$ m/s, while the gas with the density $\rho^+ = 1$ kg/m³ flows on both sides of the liquid area with very high initial velocity $v_x^+ = 30$ m/s. A spatial velocity perturbation is initialized in the internal field and has the distribution

$$v_y(y) = 0.01v_x^- \sin 2\pi \frac{x}{L} \exp -\left(\frac{2y}{h}\right)^2,$$

where h indicates the thickness of the liquid region, i.e. 0.3 mm. The simulations are tested with a resolution of $N_x \times N_y \times N_z = 256 \times 256 \times 1$. The duration of the simulation is set to

0.003 s, allowing for sufficient number of time steps for the inconsistencies to develop. To highlight the dominant impacts of the convection, the surface tension force, gravity, and viscosity are excluded from the simulations. The periodic condition is employed for all boundaries. The results are compared with the results computed by the ONERA DYJEAT codes [7, 133–136], which upholds consistent mass-momentum transport through solving the temporary density equations together with momentum equations on staggered meshes.

Figure 35 provides a quantitative comparison among multiple schemes in table 5 and with the DYJEAT code, focusing on the temporal evolution of the normalized momentum error evaluated using eq. (9.5). From the plot, it is evident that only two cases using interIsoFoam remain stable, namely, as expected using Euler + upwind, but also Euler + limitedLinearV. The zoomed-in view in fig. 35 highlights the accuracy of the results for these four stable cases. The errors calculated from DYJEAT are larger than errors from stable cases using interIsoFoam, which are around 4%. As shown in the detail in fig. 35, the errors from consistent Euler + are minimal, with respect to all other combinations of schemes.

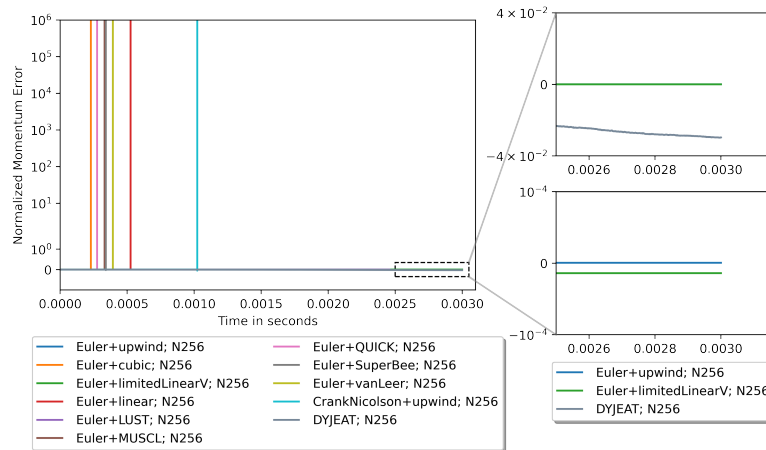


Figure 35: Time evolution of normalized momentum error of mixing layer with different schemes and DYJEAT codes, density ratio: 10^3 , resolution: $N = 256$.

9.5 Validation of a single rising bubble

In the present study, the performance of the proposed method is investigated by applying it to the simulation of a single bubble rising in a quiescent viscous liquid. To validate the proposed approach, the configuration presented by Anjos et al. [5] is adopted, who simplified the rising bubble experiments originally conducted by Bhaga and Weber [3]. In their work, Anjos et al. [5] select three distinct viscosities for comparative analysis. The focus, in particular, lies on the cases, characterized by larger liquid viscosities. The specific cases correspond to a Morton number $Mo = g\nu_l^4/\rho_l\sigma^3 = (848, 41.1, 1.31)$, where g represents the gravitational acceleration, and ν_l , ρ_l , and σ denote the viscosity, density of the ambient liquid, and surface tension coefficient, respectively. Three resolutions are tested: $N \in (64, 96, 128)$.

For the presented simulations, the air bubble is initialized with an idealized spherical shape, possessing a diameter of $D = 2.61$ cm. The air properties are defined by a viscosity of 1.78×10^{-5} kg/(m s) and a density of 1.225 kg/m³, while the liquid properties encompass viscosities of (2.73, 1.28, 0.54) kg/(m s) and a density of 1350 kg/m³. Furthermore, the surface tension between the air bubble and the liquid is 0.078 N m⁻¹. The computational domain is defined as $(-4D, -4D, -2D) \times (4D, 4D, 6D)$, where the positions of the space diagonal vertices of the computational domain are delineated, with the initial position of the bubble set at the origin, $(0, 0, 0)$. A set of dimensionless normalized variables is introduced as follows:

$$\mathbf{w} = \frac{\mathbf{v}}{\sqrt{gD}}, \quad t = \sqrt{\frac{g}{D}}\tau, \quad (9.8)$$

where τ indicates the physical time in seconds.

As illustrated in fig. 36, the utilization of the Euler and upwind schemes ensures the preservation of equivalence between volume fraction and mass advection equation. Consequently, the results obtained from `interIsoFoam` and `interIsoRhoFoam`, when employing the identical Morton number, display a substantial level of similarity. When considering $Mo = 41.1$, as demonstrated in fig. 36c and fig. 36d, the velocities acquired from both solvers demonstrate an initial increase until approximately $t = 1$, followed by a subsequent decline leading to a stable state. Notably, the differences among the results become evident when dealing with the cases involving $Mo = 41.1$. Specifically, in instances where a coarse mesh ($N = 64$) is employed, the velocity decline is more pronounced, resulting in a smaller final

stable velocity compared to the values presented in prior works [3–5]. The larger final velocity disparity from the previous works, can be observed for $Mo = 1.31$ with a mesh resolution $N = 64$ in figs. 36e and 36f, which is resulted from that the rising bubble with lower viscosities has a stronger deformation and has a thinner structure required to be captured. Using a coarse mesh loses this information and subsequently causes a larger error. Conversely, higher resolutions yield stable velocities that agree well with the experimental data documented by Bhaga and Weber [3] and Hua and Lou [4]. On the other hand, for cases with $Mo = 848$, characterized by a larger viscosity, the impact of resolution is less conspicuous, as depicted in fig. 36a and fig. 36b. The final velocities attained from different resolutions converge similarly to values that fall between the results obtained in the simulation conducted by Hua and Lou [4] and the experimental study conducted by Bhaga and Weber [3].

Furthermore, the profiles obtained by slicing the surfaces of the droplets, passing through their centers, are compared with the experimental profiles from Bhaga and Weber [3], as illustrated in fig. 37. For both solvers, the droplets with varying viscosities exhibit final shapes that closely align with the experimental visualizations, validating the hypothesis of the equivalence between Euler+upwind discretization of eq. (2.15) using the scaled mass flux $|V_f^\alpha|_s$ from eq. (8.22), and the solution of the auxiliary density equation using algorithm 6.

9.6 Liquid jet in high speed gaseous cross-flow

Different from the parallel velocities of the mixing layers in section 9.4, the liquid flows with a lower velocity is perpendicular to the velocity of the gaseous phase in this case, which is called the injection of a liquid jet in a gaseous cross-flow (LJCF) and is common in many engineering applications. The geometry and the physical properties are configured by referring to Zuzio et al. [7]. The rectangular computational domain $\Omega : [-0.01, 0, -0.01] \times [0.03, 0.02, 0.01]$ m has two inlets. The gas flows in with a velocity $\mathbf{v}^+ = [65, 0, 0]$ m/s from the left boundary x_{min} . Thuillet [0] revealed the impact of the liquid inlet velocity profile on the jet trajectory. He simulated the jet with an uniform liquid inlet velocity profile, and with a velocity profile calculated through simulating the injector. The jet trajectory results from the case with calculated velocity profile showed better agreement with the experiment. The calculated

liquid injected velocity profile from [0] is utilized, which is

$$\mathbf{v}_y^- = -21.434 \left(\frac{r}{d}\right)^3 + 15.512 \left(\frac{r}{d}\right)^2 + 8.6504$$

, where \mathbf{v}_y^- is the y -component of the liquid inlet velocity \mathbf{v}^- , r indicates the distance to the nozzle center, and d is the diameter of the nozzle. The x, z -component of \mathbf{v}^- are set to zero, whereas $d = 0.002$ m. The nozzle's center locates at $[0, 0, 0]$ in the bottom boundary patch y_{min} . To save the computation resource, the uniform Cartesian mesh with a moderate resolution of $[N_x, N_y, N_z] = [128, 64, 64]$ has been adopted. Figure 38 depicts the flow domain. The physical properties are $\rho^- = 1000$ kg/m³, $\rho^+ = 1.225$ kg/m³, $\mu^- = 1.0 \times 10^{-3}$ kg/(m s), $\mu^+ = 1.78 \times 10^{-5}$ kg/(m s), $\sigma = 7.2 \times 10^{-2}$ N m⁻¹, $g = 9.81$ m/s².

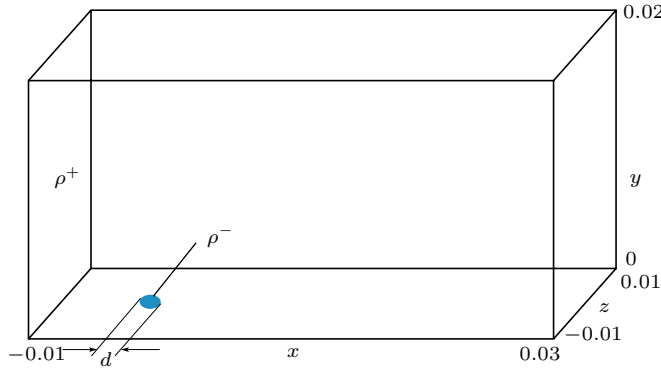



Figure 38: Liquid in a cross flow.

Figure 40 shows the final state of the injected liquid at $t = 7.1$ ms using interIsoFoam and interIsoRhoFoam with Euler and Gauss upwind regarding the iso-value of the reconstructed distance function $RDF = 0$, as well as using DYJEAT [7] with a high resolution of $[N_x, N_y, N_z] = [1024, 512, 512]$, whose shape is rendered by the iso-value 0.5 of the volume fraction. Figure 40a and fig. 40b illustrate the ruptured liquid jet from the y -side view. The droplets' distributions of interIsoFoam and interIsoRhoFoam display many comparable characteristics. There are two strips of droplets and a strip of bag-like liquid structure. The outer liquid segregates into two yz -plane symmetric strips of droplets at an early stage, i.e., at a low penetration height, as shown in the right subfigure of fig. 40a and fig. 40b. These

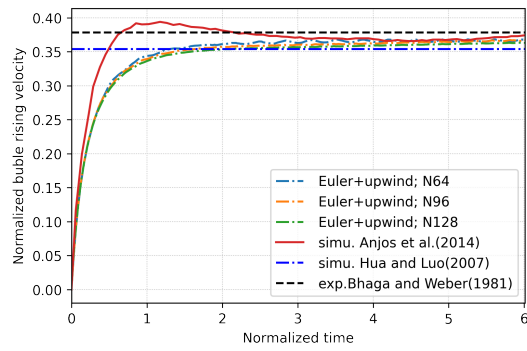
droplets translate with the gas along the x -direction and spread spanwise in the y -direction. The remaining center liquid has a wavelike detachment and forms the bag-like structure in the middle of the strips. Figure 40c demonstrates the results from DYJEAT codes with a higher resolution. Similar liquid distributions can be observed: the liquid in the center zone of the nozzle propagates like a wave and breaks up into some large packets at a higher position, whereas the liquid in the periphery of the nozzle zone rips at a low penetration height. The tests using interIsoFoam with unstable schemes are also conducted, and they fail. An example combination of Euler + cubic, is shown in fig. 54, with the simulations on both coarser and finer mesh fail catastrophically.

This validation case is a candidate for a benchmark case for validating two-phase flow numerical methods that consistently handle high density ratios, because the experimental form of the jet can be accurately reproduced on coarser mesh resolutions. Using a coarser mesh resolution interIsoFoam and interIsoRhoFoam of course do not capture the small structures such as liquid streaks, sacs and droplets, as shown in DYJEAT's results. However, the solvers accurately predict the jet curve, which can be used as a quantifiable argument for validity of a consistent method against experimental data.

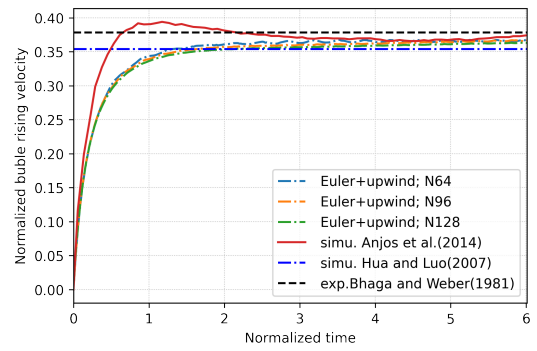
Figure 39 displays the final bent shape of liquid jet simulated by three solvers and their comparison with the experimental observation made by ONERA [0]. The same case with two different resolutions are tested. The liquid jets' shape results in the same parallel view are compared with each other and also with the experimental results marked by the red line. The blue translucent liquid jet represents the results from the case with a higher resolution $N_h = [254, 128, 128]$, while the gray liquid jet comes from the above low-resolution results. It can be seen from each figs. 39a to 39c that more small droplets and complex structures can be captured when deploying the higher mesh resolution. Despite the different resolutions, there exists a very minor difference between the liquid jets with regard to the bent shape. The windward surfaces of the two liquid jets in each sub-figure almost attach to each other, which highlights that this case is insensitive to the mesh resolution. As to the comparison with the experimental trajectories, both the jets' bent surface in figs. 39a and 39b show good correspondence to the experimental shape, i.e. the red line at the low penetration height < 9 mm. The jets reattach to the red line in the upper-right zone. The maximal deviation between the simulated jets and the experiment shape is around 1 mm, which is 5% of the jet



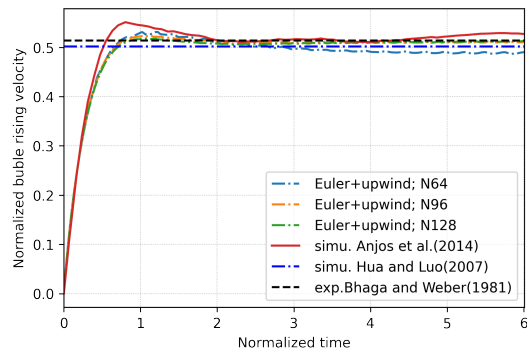
height. A more obvious discrepancy between the jets and the experiment is shown in fig. 39c. The liquid jet bent less than the experiment in the high-speed flow after the given time, which results in a wrong prediction of the impingement position.



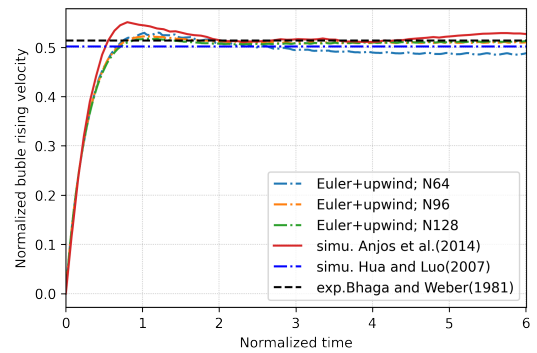
(a) interIsoFoam: $Mo = 848$



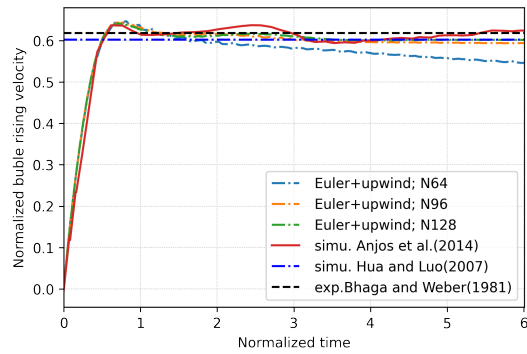
(b) interIsoRhoFoam: $Mo = 848$



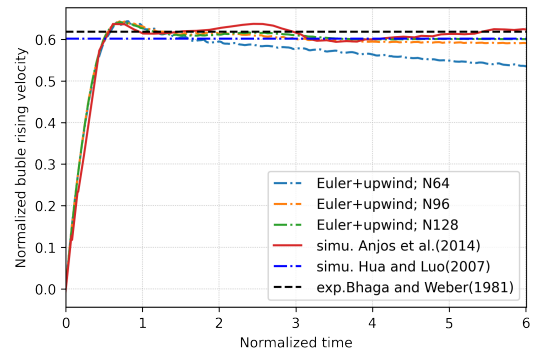
(c) interIsoFoam: $Mo = 41.4$



(d) interIsoRhoFoam: $Mo = 41.4$

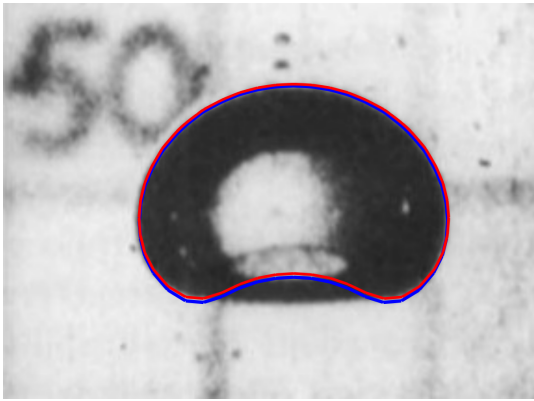


(e) interIsoFoam: $Mo = 1.31$

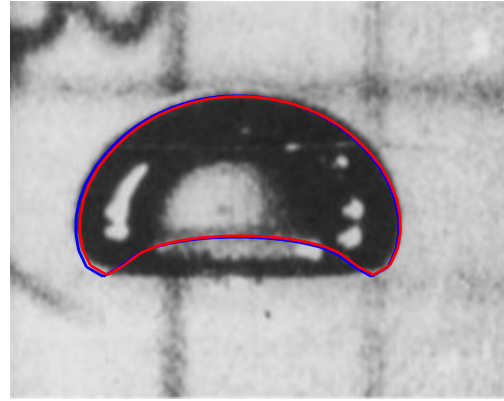


(f) interIsoRhoFoam: $Mo = 1.31$

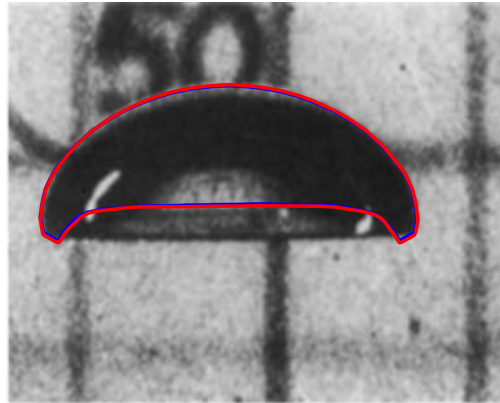
Figure 36: Temporal evolution of rising velocity using interIsoFoam and interIsoFoam: Euler + upwind, $\rho^- / \rho^+ \approx 10^3$.



(a) $Mo = 848$



(b) $Mo = 41.4$



(c) $Mo = 1.31$

Figure 37: Comparisons of final shapes of rising bubbles using interIsoFoam and interIsoRhoFoam with the experimental visualization from Bhaga and Weber [3] (reprinted with permission): Euler + upwind, red line from interIsoRhoFoam, blue line from interIsoFoam, $\rho^-/\rho^+ \approx 10^3$, $N = 128$, $t = 6$.

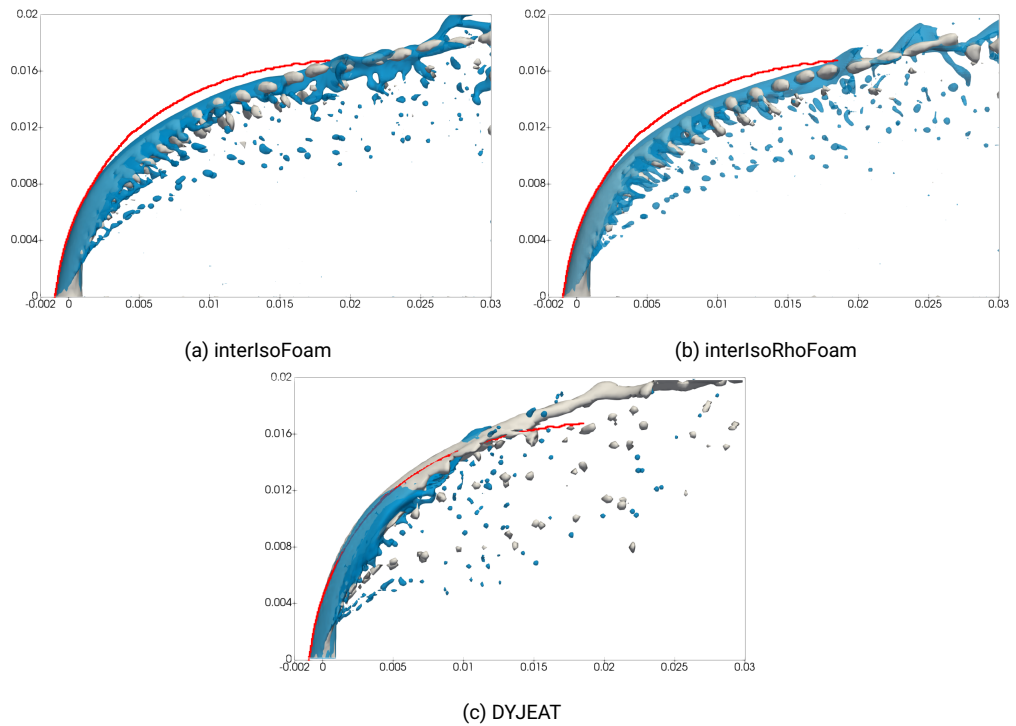
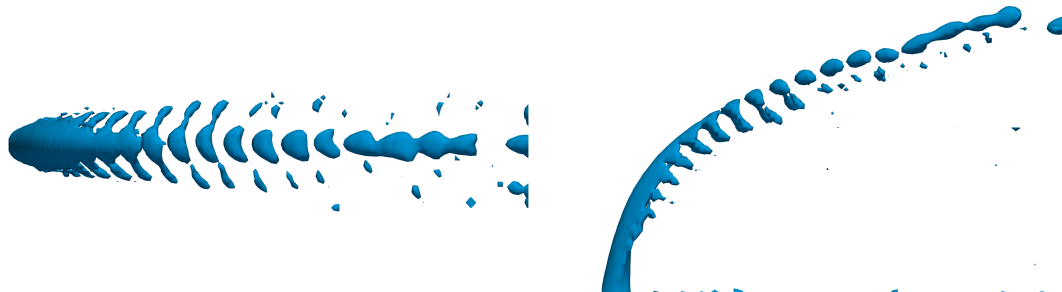


Figure 39: The instantaneous liquid jet shape at final time $t = 7.1ms$ with different resolutions (the blue translucent jet: $N_h = [256, 128, 128]$; the gray jet: $N_l = [128, 64, 64]$) and its comparison with the experimental results(the red line). This case makes it possible to evaluate performance on coarser meshes as resolving finer structures does not impact the jet trajectory.



(a) interIsoFoam: $t = 7.1$ ms



(b) interIsoRhoFoam: $t = 7.1$ ms



(c) DYJEAT: $t = 7.07$ ms

Figure 40: The shape of the injected liquid with interIsoFoam and interIsoRhoFoam (Euler and Gauss upwind, density ratio: 816, CFL number: $CFL = 0.2$, resolution: $N_t = [128, 64, 64]$), and with DYJEAT (resolution: $N = [1024, 512, 512]$ [7])

A residual-based non-orthogonality correction for force-balanced unstructured Volume-of-Fluid methods

10 Introduction

Numerical methods for simulating two-phase flows must ensure a balance of forces acting on the fluid interface on the discrete level. An imbalanced discretization makes it impossible in some cases to achieve a steady-state interface shape, e.g. for a canonical case of a spherical droplet suspended in air without the influence of gravity, or a stationary droplet that is wetting a surface and stationary liquid column in equilibrium characterized by high density ratio.

Non-orthogonality errors in unstructured Finite Volume methods for simulating incompressible two-phase flows may break the force-balanced discretization. It is shown in this chapter that applying the same explicit non-orthogonality correction for all gradient terms in the context of segregated solution algorithms is not sufficient to achieve force balance. To ensure force balance, a straightforward and deterministic residual-based control of the non-orthogonality correction is introduced, which removes the number of non-orthogonality corrections as a free parameter from the simulation. This method is directly applicable to different unstructured finite-volume two-phase flow simulation methods as long as they discretize the one-field formulation of incompressible two-phase Navier-Stokes equations. The demonstration of force balance for the surface tension force and the gravity force near linear solver tolerance for an algebraic and a geometric Volume-of-Fluid method is provided

in this chapter using the stationary droplet and stationary water column verification cases on polyhedral unstructured meshes with varying levels of non-orthogonality.

The force-balanced discretization in the unstructured Finite Volume Method [19, 28, 137] is, because of its high degree of volume conservation and its ability to discretize boundary conditions at geometrically complex domain boundaries with second-order accuracy. This approach is implemented and verified it for the unstructured geometrical VOF method [6] and the unstructured algebraic VOF method [138].

In section 11, the force-balanced discretization and their solution algorithm is described in details. In section 11, the same principle of error cancellation in the structured force-balanced discretization [139] intuitively extends to unstructured meshes, is demonstrated, contrary to recent findings in [140], under the condition that the pressure and velocity equation solution converges. In section 11.3, with the knowledge about the mathematical model and the unstructured Finite-Volume and VOF discretization from sections 8 and 11, respectively, the proposed approach is compared with state-of-the-art methods.

11 Methodology review

In this chapter, the impact of the UFVM on the balance of forces at the fluid interface is investigated when the UFVM meshes are non-orthogonal and a highly accurate, computationally efficient and deterministic stopping criterion is developed for the iterative non-orthogonality correction in the UFVM. Handling non-orthogonality in UFVM is crucial for simulating two-phase flows in geometrically complex domains whose unstructured finite discretization results in a larger degree of non-orthogonality.

The proposed non-orthogonality correction is implemented into the unstructured geometrical VOF method [34], specifically the plicRDF-isoAdvector method [6], which is described in section 6.4. The research software [0, 141] and research data [0] are publicly available.

11.1 Solution algorithm

Discretizing volume and momentum transport equation, i.e., eqs. (2.2) and (2.15), with the implicit UFVM results in an algebraic equation system

$$\sum_{f \in F_c} F_f^{n+1} = 0, \quad (11.1)$$

$$a_c \mathbf{v}_c^{n+1} + \sum_{n \in N} a_n \mathbf{v}_n^{n+1} = -(\nabla p)_c^{n+1} - ((\mathbf{g} \cdot \mathbf{x}) \nabla \rho)_c^{n+1} - \sigma_c (\kappa \nabla \alpha)_c^{n+1} + \mathbf{S}_c(\mathbf{v}_c^n), \quad (11.2)$$

solved together with the unstructured geometric VOF equation eq. (8.14). In eq. (11.2), a_c is the linear equation system coefficient of the finite volume Ω_c , N is the set of finite volumes that are face-adjacent to Ω_c , and $\mathbf{S}_c(\mathbf{v}_c^n)$ is the momentum source term containing contributions from all operators in eq. (2.15) from time t^n . The details of the derivation for eq. (11.2) are referred to section 3.

Equations (8.14), (11.1) and (11.2) cannot be solved simultaneously at the new time step t^{n+1} , because eqs. (11.1) and (11.2) are linear algebraic equations resulting from an implicit UFVM discretization, and eq. (8.14) is solved geometrically. To ensure that equations eqs. (8.14), (11.1) and (11.2) are satisfied at t^{n+1} , a segregated solution algorithm developed by OpenFOAM[®] is selected as basis, i.e. PIMPLE, as discussed in section 3.4, to solve these equations, sequentially iterating until all equations are satisfied.

A simplified description of the PIMPLE algorithm is adjusted from section 3.4 here, focusing on the interplay between the curvature approximation accuracy, surface tension force approximation and mesh non-orthogonality as the primary sources of numerical instabilities.

Dividing semi-discrete momentum equation eq. (11.2) by the coefficients a_c , and applying Rhie-Chow interpolation [142] gives

$$\mathbf{v}_f^i = - \left(\frac{1}{a_c} \right)_f (\nabla p)_f^i - \left(\frac{1}{a_c} \right)_f ((\mathbf{g} \cdot \mathbf{x}) \nabla \rho)_f^o - \left(\frac{1}{a_c} \right)_f \sigma_f (\kappa \nabla \alpha)_f^o + \left(\frac{1}{a_c} \right)_f (\mathbf{H}(F_f^o, \mathbf{v}^{i-1}))_f, \quad (11.3)$$

where o denotes the outer iteration in which the momentum equation is discretized (and solved), and i denotes the inner iterations used for solving the pressure Poisson equation derived below, and $\mathbf{H}(F_f^o, \mathbf{v}^{i-1}) := - \sum_{n \in N} a_n \mathbf{v}_n^{i-1} + \mathbf{S}_c(\mathbf{v}^n)$, with a_n containing, among other

terms, F_f^o , the volumetric flux resulting from the integration of eq. (8.2) with second-order accurate face-centered quadrature $F_f^o = \mathbf{v}_f^o \cdot \mathbf{S}_f$.

Applying eq. (11.1) to eq. (11.3), leads to the discrete Poisson equation for the pressure

$$\begin{aligned} \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f (\nabla p)_f^i \cdot \mathbf{S}_f &= - \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f (\mathbf{g} \cdot \mathbf{x})_f (\nabla \rho)_f^o \cdot \mathbf{S}_f \\ &\quad - \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f \sigma_f \kappa_f^o (\nabla \alpha)_f^o \cdot \mathbf{S}_f \\ &\quad + \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f \mathbf{H}(F_f^o, \mathbf{v}^{i-1})_f \cdot \mathbf{S}_f. \end{aligned} \quad (11.4)$$

The volume fraction, velocity, and pressure are coupled in eqs. (8.14), (11.3) and (11.4), respectively, with the volume fraction α_c^o available from the solution of eq. (8.14) using the plicRDF-isoAdvector method, or any other method that approximates $\chi(\mathbf{x}, t)$ as a discrete phase-indicator α in the form of volume fractions, or, similarly, a marker field [47].

11.2 Force Balance

Regarding $(\nabla p)_f^i$, $(\mathbf{g} \cdot \mathbf{x})_f (\nabla \rho)_f^o$ and $\sigma_f \kappa_f^o (\nabla \alpha)_f^o$: it is widely known that all face-centered gradients must be discretized with the same scheme to ensure a force-balanced discretization, that can be shown for the balanced CSF model [139].

To show that the Unstructured Finite Volume discretization is also balanced, a sphere suspended in zero-gravity with a constant mean curvature κ in a steady state is investigated, which reduces eq. (11.4) to

$$\sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f^o (\nabla p)_f^i \cdot \mathbf{S}_f = - \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f^o \sigma_f \kappa (\nabla \alpha)_f^o \cdot \mathbf{S}_f \quad (11.5)$$

i.e.,

$$(\nabla p)_f^i + \sigma_f \kappa (\nabla \alpha)_f^o = 0, \quad (11.6)$$

and if the same discretization scheme $(\nabla \cdot)_f$ is used, then following holds for a sphere

$$\nabla(p^i + \sigma \kappa \alpha^o) = 0, \quad (11.7)$$

finally leading to

$$p^i = p^{C,i} - \sigma \kappa^o \alpha^o, \quad (11.8)$$

where $p^{C,i}$ is any constant pressure. Equation (11.8) recovers the exact Young-Laplace pressure-jump across the interface $\Sigma(t)$, equivalently to structured discretizations [139]. If the curvature as κ^a is approximated with some approximation error ϵ_κ , i.e., $\kappa^a = \kappa + \epsilon_\kappa$ and balance the forces using

$$(\nabla p)_f^i + \sigma_f \kappa_f^a (\nabla \alpha)_f = 0, \quad (11.9)$$

while considering only orthogonal gradient discretization, this results in

$$\begin{aligned} \nabla(p + \sigma \kappa \alpha) &= -\sigma \epsilon_\kappa (\nabla \alpha)_f, \\ \frac{p'_{N_f}{}^i - p'_{O_f}{}^i}{|\mathbf{d}_f|} &= -\frac{\sigma \epsilon_\kappa (\alpha_{N_f}^o - \alpha_{O_f}^o)}{|\mathbf{d}_f|}, \\ p'_{N_f}{}^i &= p'_{O_f}{}^i - \sigma \epsilon_\kappa (\alpha_{N_f}^o - \alpha_{O_f}^o), \end{aligned} \quad (11.10)$$

where $p' = p + \sigma \kappa \alpha$, and O_f, N_f are the indices of cell centers from cells adjacent to the face \mathbf{S}_f . Equation (11.10) shows that modification of the Young-Laplace pressure, ensured for constant κ by eq. (11.8) in the order-of-accuracy of $\nabla(\cdot)_f$, is linearly proportional to the curvature approximation error ϵ_κ , making curvature approximation crucial.

Curvature approximation in the context of numerical methods for two-phase flows is a long-standing challenge, which is not addressed here. Instead, here the focus is put on ensuring that the unstructured Finite Volume Method (FVM) remains force-balanced on non-orthogonal meshes that are ubiquitous to industrially relevant two-phase flow problems that have geometrically complex flow domains.

11.3 Force Balance on Non-Orthogonal Meshes

The unstructured Finite Volume Method must retain second-order accuracy and force balance also on non-orthogonal meshes: where the face area-normal vector \mathbf{S}_f is not collinear with the vector connecting the centroids of cells, that share \mathbf{S}_f , i.e., $\mathbf{d}_f := \mathbf{x}_{N_f} - \mathbf{x}_{O_f}$, as depicted in fig. 41. A widely used approach for non-orthogonality correction uses the principle of error superposition, i.e. splitting the total gradient flux into an explicit non-orthogonal \parallel and

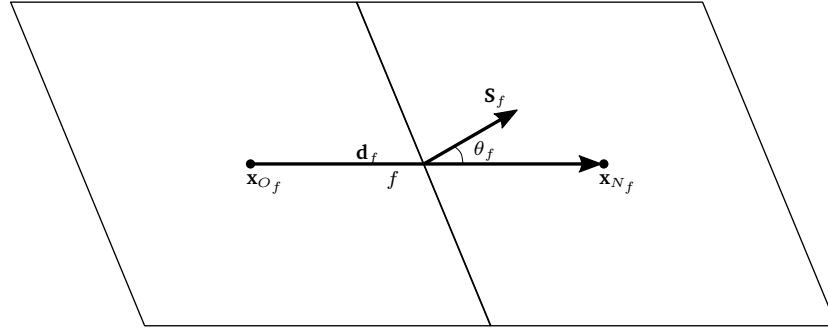


Figure 41: Representation of non-orthogonality: $\mathbf{x}_{O_f}, \mathbf{x}_{N_f}$ are the centroids of two adjacent cells O, N ; \mathbf{d}_f is the vector connecting \mathbf{x}_{O_f} and \mathbf{x}_{N_f} .

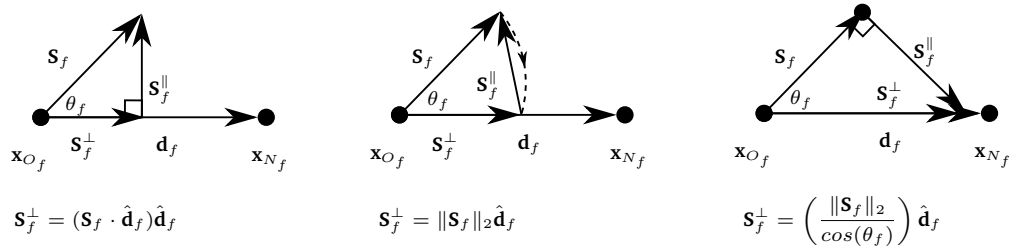


Figure 42: Three common non-orthogonality correction approaches: minimum correction (left), orthogonal correction (middle), over-relaxed correction(right). Vector $\hat{\mathbf{d}}_f$ is the unit vector of \mathbf{d}_f , i.e., $\hat{\mathbf{d}}_f := \frac{\mathbf{d}_f}{|\mathbf{d}_f|}$.

implicit orthogonal \perp contribution. Several non-orthogonal corrections [28, 137, 143, 144],

$$(\nabla\phi)_f \cdot \mathbf{S}_f = (\nabla\phi)_f^\perp \cdot \mathbf{S}_f^\perp + (\nabla\phi)_f^\parallel \cdot \mathbf{S}_f^\parallel, \quad (11.11)$$

for a property ϕ , as shown in fig. 42. A runtime-configurable simulation software such as OpenFOAM® [19] enables very straightforward consistent gradient scheme selection at the start of a simulation.

Following the principle of error superposition, if $(\nabla p)_f^\parallel \cdot \mathbf{S}_f^\parallel$ balances out $\sigma_f \kappa_f (\nabla\alpha)_f^\parallel \cdot \mathbf{S}_f^\parallel$ on the right hand side of eq. (11.5), after applying eq. (11.11) in eq. (11.5) to p and α , the non-orthogonality correction will be force-balanced. The logic following the force-balanced orthogonal gradient is that the same scheme used to discretize $(\nabla p)_f^\parallel \cdot \mathbf{S}_f^\parallel$ and $\sigma_f \kappa_f (\nabla\alpha)_f^\parallel \cdot \mathbf{S}_f^\parallel$ will ensure force-balance.

However, ensuring that all terms are corrected in the same way is insufficient for force balance in the context of two-phase flow simulations. Since the non-orthogonality correction is explicit, $\sigma_f \kappa_f (\nabla\alpha)_f^\parallel \cdot \mathbf{S}_f^\parallel$ lags behind internal iterations in eq. (11.4). Focusing only on the force balance between the pressure gradient and the surface tension force in eq. (11.4), an explicit non-orthogonality correction becomes

$$\begin{aligned} \sum_{f \in F_c} \left(\frac{1}{a_c}\right)_f^o (\nabla p)_f^{k,\perp} \cdot \mathbf{S}_f^\perp &= - \sum_{f \in F_c} \left(\frac{1}{a_c}\right)_f^o \sigma_f \kappa_f^o (\nabla\alpha)_f^{o,\perp} \cdot \mathbf{S}_f^\perp \\ &\quad - \sum_{f \in F_c} \left(\frac{1}{a_c}\right)_f^o \sigma_f \kappa_f^o (\nabla\alpha)_f^{o,\parallel} \cdot \mathbf{S}_f^\parallel \\ &\quad - \sum_{f \in F_c} \left(\frac{1}{a_c}\right)_f^o (\nabla p)_f^{k-1,\parallel} \cdot \mathbf{S}_f^\parallel. \end{aligned} \quad (11.12)$$

In eq. (11.12), the explicit non-orthogonality correction indexed with $k - 1$ lags behind the implicit orthogonal pressure gradient indexed with k in balancing orthogonal and non-orthogonal $(\nabla\alpha)_f \cdot \mathbf{S}_f$ from the outer iteration o . This lag, if not resolved by absolutely ensuring a sufficient number of k iterations are applied, causes oscillations in the pressure and, in turn, large parasitic velocities on non-orthogonal meshes, which perturb the fluid interface (volume fractions), reflecting the errors further into the curvature approximation, which closes the loop by forwarding the errors again into the pressure through eq. (11.4).

A deterministic stopping criterion for the k non-orthogonality iteration is proposed in

this work, which ensures force-balance for non-orthogonal meshes that avoids introducing a problem-specific number of corrections, and significantly modifying the UFVM, or the solution algorithm. The solution is straightforward and becomes clear if the linear system given by eq. (11.4), extended with the non-orthogonality correction from eq. (11.12), is written in matrix form as

$$\mathbf{L}^o \mathbf{p}^k = \mathbf{b}^o + \mathbf{s}_{no}^{k-1} \quad (11.13)$$

where \mathbf{p} (pressure), \mathbf{b} (source term), \mathbf{s}_{no} (non-orthogonality source) are cell-centered fields of the size equal to the number of cells in the mesh ($|C|$), and

$$\mathbf{s}_{no}^{k-1} = - \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f^o \sigma_f \kappa_f^o (\nabla \alpha)_f^{o, \parallel} \cdot \mathbf{s}_f^{\parallel} - \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f^o (\nabla p)_f^{k-1, \parallel} \cdot \mathbf{s}_f^{\parallel} \quad (11.14)$$

is the force-imbalance between the pressure gradient and the surface-tension force resulting from the explicit non-orthogonal correction. The source term \mathbf{s}_{no} should diminish with increasing k , independent of the chosen non-orthogonality correction, as long as the same correction is applied to all gradients, and the correction is, of course, convergent. A deterministic way to ensure that \mathbf{s}_{no}^k diminishes, is by

$$|\mathbf{L}^o \mathbf{p}^k - \mathbf{b}^o - \mathbf{s}_{no}^{k-1}|_{\lambda} < \tau, \quad (11.15)$$

i.e., requiring the non-orthogonal force-balance to reach the accuracy of the linear-solver tolerance τ_S using a linear-solver residual norm λ_S , for the pressure Poisson equation. Equation (11.15) works because of the way modern linear solver algorithms are implemented. Given that, in Computational Fluid Dynamics, PDEs whose solutions evolve either over iterations for steady state problems, or time steps for transient problems, in each subsequent iteration, are solved, the linear solver will first compute the initial residual using the solution from the previous iteration. If the chosen non-orthogonality correction is convergent, with increasing k , it will achieve force-balance. At the point of force-balance, the linear solver will use the current \mathbf{p}^k to compute the residual of eq. (11.4), which will fall under the tolerance τ given the solver norm λ . This defines a straightforward and entirely deterministic termination criterion for a force-balanced explicit non-orthogonal correction, and does so, without the need to quantify the tolerance for the non-orthogonality error. In addition to being deterministic, the residual-based non-orthogonality correction in eq. (11.15) works in

the same way for any linear solver and any tolerance. Equation (11.15) will incur a minimal number of corrections - the force-balance cannot be more accurate than satisfying eq. (11.4) using the chosen linear solver norm and tolerance. If the non-orthogonality error is small, \mathbf{s}_{no}^{k-1} will diminish quickly, if it is large, an increase of iterations is justified and confirmed by our results section.

Satisfying eq. (11.15) is very straightforward in a numerical source code: it is equivalent to expecting that the linear solver exits the initial iteration since this means that the initial p^k used to compute the initial residual, already satisfies eq. (11.15).

The proposed non-orthogonality correction is designated as Residual-based Non-Orthogonality Correction (*ResNonOrthCorr*) and outline the required straightforward modification of the segregated solution algorithm to incorporate ResNonOrthCorr in algorithm 9.

In algorithm 9, the option is added to stop if the number of iterations exceeds N_{max} as means of avoiding exceedingly large number of corrections that would lead to computationally intractable simulations, in cases with unavoidable extremely large non-orthogonality in industrial applications.

When adding the gravity force to balanced forces, equivalently to the CSF surface tension force, the same principle of error-superposition is applied and \mathbf{s}_{non}^k is extended to include the explicit non-orthogonal gravity force contribution

$$\begin{aligned} \mathbf{s}_{no}^{k-1} = & - \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f^o [\sigma_f \kappa_f^o + (\rho^- - \rho^+) (\mathbf{g} \cdot \mathbf{x})_f] (\nabla \alpha)_f^{o, \parallel} \cdot \mathbf{S}_f^{\parallel} \\ & - \sum_{f \in F_c} \left(\frac{1}{a_c} \right)_f^o (\nabla p)_f^{k-1, \parallel} \cdot \mathbf{S}_f^{\parallel}. \end{aligned} \quad (11.16)$$

It is important to note at this point that, from eq. (2.10),

$$\nabla \rho = (\rho^- - \rho^+) \nabla \chi \approx (\rho^- - \rho^+) \nabla \alpha. \quad (11.17)$$

With sufficient information about the UFVM discretization, segregated solution algorithms and non-orthogonal force balance, ResNonOrthCorr can be verified.

Comparison with state-of-the-art methods

The ResNonOrthCorr is first compared with the widely used heuristic fixed-number of non-orthogonality corrections (*FixNonOrthCorr*) in the PIMPLE algorithm 8 algorithm in OpenFOAM [145] and similar iterative segregated solution algorithms.

The PIMPLE method updates α_c , \mathbf{v}_c and p_c as outlined in algorithm 8. In each outer loop, the geometric information of the interface and physical properties are updated. In the inner loop, source terms of eq. (11.4) are calculated and eq. (11.4) is solved.

Algorithm 8 FixNonOrthCorr in the PIMPLE solution algorithm.

```

1: while  $t \leq t_{end}$  do
2:    $t^{n+1} = t^n + \Delta t$ 
3:   for  $o = 1; o \leq N_{outer}; ++o$  do
4:     Reconstruct the fluid interface, i.e.,  $\tilde{\chi}(\mathbf{x}, t^{o-1}) \approx \chi(\mathbf{x}, t^{o-1})$  from  $\alpha_c^{o-1}$ .
5:     Solve eq. (8.14) for  $\alpha_c^o$  using  $\tilde{\chi}(\mathbf{x}, t^{o-1}), F_f^{o-1}$ , giving  $\rho_{f,c}^o, \mu_f^o$  from eqs. (2.10) and (2.11), respectively.
6:     Discretize the momentum eq. (11.2) using  $\alpha_c^o, \rho_c^o, \rho_f^o, \mu_f^o$  compute the  $\mathbf{H}(F_f^{o-1}, \mathbf{v}^{o-1})$  operator.
7:     for  $i = 1; i \leq N_{inner}; ++i$  do
8:       for  $k = 1; k \leq N_{non}; ++k$  do ▷ Non-orthogonality correction.
9:         Solve the pressure equation eq. (11.4) for  $p_k^i$  with  $\mathbf{H}(F_f^o, \mathbf{v}^{i-1})$  and  $\alpha_c^o, \rho_c^o$ .
10:      end for
11:      Update  $F_f^i, \mathbf{v}_c^i$  with  $\mathbf{H}(F_f^o, \mathbf{v}^{i-1})$  and  $p_c^{Non}$  using
12:    end for
13:  end for
14: end while

```

Correcting face-centered gradients for non-orthogonality introduces the non-orthogonality correction loop k in algorithm 8. Generally, N_{outer} , N_{inner} and N_{non} , are set by the user of algorithm 8, making N_{non} a problem-specific "free" parameter. The number N_{non} in algorithm 8 requires adjustment by trial and error; It is shown in the results section that this is insufficient for achieving force-balance. It is widely known that the pressure Poisson eq. (11.4) is the computational bottleneck in CFD. In segregated solution algorithms, the computational cost is directly proportional to the number of times eq. (11.4) is solved, namely,

$N_{outer} \times N_{inner} \times N_{non}$. The ResNonOrthCorr completely removes N_{non} as a "free" parameter from the solution algorithm 8 and replaces the stopping condition in the innermost loop in algorithm 9.

Algorithm 9 ResNonOrthCorr in the PIMPLE algorithm.

```

1: while  $t \leq t_{end}$  do
2:    $t^{n+1} = t^n + \Delta t$ 
3:   for  $o = 0; o < N_{outer}; ++ o$  do
4:     Reconstruct the fluid interface, i.e.,  $\tilde{\chi}(\mathbf{x}, t^{o-1}) \approx \chi(\mathbf{x}, t^{o-1})$  from  $\alpha_c^{o-1}$ .
5:     Solve eq. (8.14) for  $\alpha_c^o$  using  $\tilde{\chi}(\mathbf{x}, t^{o-1}), F_f^{o-1}$ , giving  $\rho_{f,c}^o, \mu_f^o$  from eqs. (2.10) and (2.11), respectively.
6:     Discretize the momentum eq. (11.2) using  $\alpha_c^o, \rho_c^o, \rho_f^o, \mu_f^o$  compute the  $\mathbf{H}(F_f^{o-1}, \mathbf{v}^{o-1})$  operator.
7:      $k = 0$ 
8:     for  $i = 0; i < N_{inner}; ++ i$  do
9:       while  $|L^0 p^{k+1} - b^o - \mathbf{s}_{no}^k|_\lambda > \tau$  or  $k > N_{max}$  do
10:         $k = k + 1$ 
11:        Solve the pressure equation eq. (11.4) for  $p_k^i$  with  $\mathbf{H}(F_f^{i-1}, \mathbf{v}^{i-1})$  and  $\alpha_c^o, \rho_c^o$ .
12:       end while
13:       Update  $F_f^o, \mathbf{v}_c^o$  with  $F_f^i, \mathbf{v}_c^i$  from  $p_k^i$  using eq. (11.3).
14:     end for
15:   end for
16: end while

```

A recent algorithm proposed by [140] seeks to balance $(\nabla p)_f$ with $(\nabla \alpha)_f$ on a non-orthogonal mesh by employing the same scheme to discretize pressure and forces gradients. This approach combines explicit correction terms for both discretized pressure and forces gradients into a "revised" pressure gradient. The reconstruction of this new pressure gradient at the cell center is accomplished using, what the authors call, the Time-evolution Converting (TEC) operator, instead of the conventional Green Gauss and least-square methods. The TEC operator, proposed in [146], is actually a variant of the least-square method that minimizes the sum of squares of the error between the computed fluxes from the cell center gradient and the directly estimated fluxes at the cell faces. This has been in widespread application in OpenFOAM® for reconstructing cell center values from known fluxes, as discussed by Tolle,

Bothe, and Marić [2] and Aguerre et al. [147] and revisited by Assam and Natarajan [148].

Prior works, such as those by [72, 73, 110, 146], have also acknowledged the impact of consistent discretization between pressure gradient flux and force gradient fluxes in preserving force balance on non-orthogonal meshes. Consequently, they utilized identical discretized formulations for all these gradient fluxes. However, the correction for non-orthogonality in these methods occurs only once per internal iteration when solving the discretized Poisson equation, i.e., during the assembly of the coefficients matrix and sources of the discretized Poisson equation.

The discretization of gradient flux (eq. (11.11)) has been extensively discussed, not only for pressure and force gradients but also for diffusive fluxes in various fields such as velocity, temperature, and electric fields, among others. Over the past decades, numerous methods have been developed to address this discretization challenge [144]. Muzaferija and Gosman [143] and Muzaferija [149] initiated an approach to approximate the gradient flux by interpolating the gradients at two adjacent cell centers to the intersected face and multiplying the interpolated face gradient by the face area vector. However, they found that this simple gradient interpolation method could introduce unphysical oscillations in the solution on a collocated mesh. To mitigate this issue, recoupling terms, inspired by oscillatory pressure handling methods proposed by Rhie and Chow [142], were added to the interpolated face gradient [143, 149]. Demirdžić and Muzaferija [150] made slight adaptations to these added terms [143, 149] to ensure their vanishing when the solution converges. Nishikawa [151, 152] proposed a general principle for discretizing the diffusion term based on a first-order hyperbolic system. In the context of the finite volume method, the diffusion discretization scheme involves the addition of an arithmetic average of neighboring cell center gradients and a jump term containing a free parameter. This parameter is used to dampen high-frequency errors caused by the arithmetic average. In addition to these pure numerical derivations for diffusion discretization, some works ([16, 28, 137]) chose a geometric perspective. Jasak [28] decomposed the face area vector \mathbf{S}_f into the direction of the line connecting two neighboring cell centers, denoted as \mathbf{d}_f in fig. 42, and the vector $\mathbf{S}_f - \mathbf{d}_f$. The central differencing formula was then used to implicitly approximate the product of the face gradient and the \mathbf{d}_f component, while face interpolation of the center gradient was used to explicitly approximate the product of the face gradient and the $\mathbf{S}_f - \mathbf{d}_f$ component.

The author also compared these three correction methods in fig. 42 and showed the over-relaxation correction has the best performance. Ferziger, Perić, and Street [16] extrapolated two neighboring cell center values to two auxiliary nodes, which are located on the line in the face normal direction and passing through the face center. The extrapolated values and positions of these two nodes are used to construct a central difference to approximate the face gradient. Similarly, Darwish, Mangani, and Moukalled [137] introduced a scheme utilizing two auxiliary nodes, although these two nodes are not placed on the normal line through face center. Instead, they are constrained by constructing two opposite directional gradient fluxes with the corresponding near cell centers. The final gradient flux is calculated by averaging the two fluxes. Darwish, Mangani, and Moukalled [137] indicated that the gradient flux could be fully implicit by carefully selecting the averaging parameters concerning the node values and positions. For this fully implicit scheme, each internal iterative step in solving the discretized equation contained one non-orthogonal correction, leading to fewer iterations required to converge to a satisfactory tolerance compared with the semi-implicit schemes, where the correction is conducted only once during the assembly of the source term of the discretized equation. While numerous methods have been developed, review papers such as [144, 153] reveal equivalences between the final formulations of the discretized diffusion flux in different works, e.g., [28] and [143, 149, 150]. A comprehensive comparison of various semi-implicit diffusion discretization schemes, considering both discretization and truncation errors, is presented in Jalali, Sharbatdar, and Ollivier-Gooch [154]. The authors [154] concluded that the most accurate approximation for diffusive fluxes is achieved by adding a solution jump term to the average of two adjacent cell gradients, as proposed in [151, 152].

All contemporary methods require a number of iterations for the non-orthogonality correction that is left as a free parameter. Our contribution to non-orthogonality correction using the gradient-flux decomposition, regardless of the chosen gradient scheme or the numerical method for tracking fluid interfaces, is a deterministically controlled number of iterations that ensures force-balance for two-phase flow simulations on non-orthogonal meshes, verified in the next section.

12 Results

Data archives including the algorithm implementation, input data, post-processing software and secondary data are publicly available [0]. The proposed method is actively developed in a publicly available git repository [141].

In this section, the efficiency and accuracy of the proposed force-balanced algorithm are assessed using two canonical verification tests, namely, the stationary droplet and the stationary water tank, used, respectively, to test the force-balanced surface tension force and gravity force discretization. These tests are conducted on unstructured meshes and the non-orthogonality corrections are either kept fixed (*FixNonOrthCorr*) or are controlled by our residual-based algorithm (*ResNonOrthCorr*). To systematically investigate the influence of different mesh types on the solutions, simulations are performed on equidistant unstructured mesh (*blockMesh*), perturbed equidistant unstructured mesh, i.e. hexahedral mesh (*perturbMesh*), and polyhedral mesh (*polyMesh*), as illustrated in fig. 43. Three distinct mesh resolutions are investigated, with resolutions $\Delta x \in (L/30, L/60, L/90)$. Here, L indicates the characteristic length of the cubic computational domain Ω . These hydrodynamic test cases consider a water/air fluid pairing to emphasize a problematic two-phase flow scenario, the fluid properties are outlined in table 8.

Both tests share identical termination times and fixed time steps, specifically $t_{end} = 0.01$ s and $\Delta t = 1e-4$ s. For the segregated solution algorithm algorithm 9 used in this work, exemplary, $N_{outer} = 4$, $N_{inner} = 1$. For *FixNonOrthCorr*, non-orthogonality is corrected N_{non} times within an inner loop. In the case of *ResNonOrthCorr*, the non-orthogonality correction is performed until eq. (11.15) is satisfied. The linear solver tolerance for eq. (11.4) is uniformly set to $1e-12$ for both algorithms.

12.1 Stationary droplet in equilibrium

In accordance with the Young – Laplace law, the velocity of a spherical droplet in equilibrium, in the absence of gravity, is zero, i.e. $\mathbf{v} = 0$. This is attributed to the balance between the surface tension force and the pressure jump across the interface. This test involves a stationary water droplet, with microfluidic dimensions, in equilibrium with surrounding

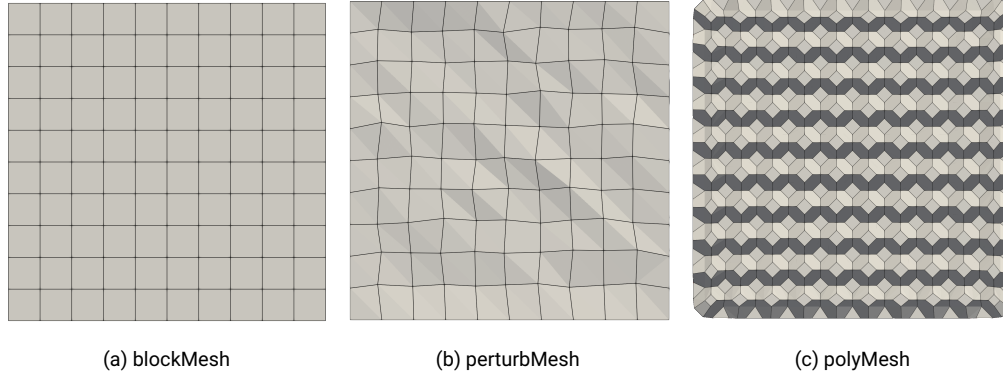


Figure 43: A sliced cell layer from a cubic computational domain with three mesh types: equidistant mesh (*blockMesh*), perturbed hexahedral mesh (*perturbMesh*) and polyhedral mesh (*polyMesh*).

20 °C	density ρ (kg/m ³)	kin. viscosity ν (m ² /s)	dyn. viscosity μ (Pa s)	surface tension σ (N/m)	gravity \mathbf{g} (m/s ²)
water[155, 156]	998.2	1e-6	9.982e-4	72.74e-3[157]	(0, 0, -9.81)
air[156]	1.19	1.53e-5	1.8207e-5		

Table 8: Physic properties of water/air pair.

air. The water droplet of radius $R = 1$ mm is placed at the centroid of the cubic domain Ω , with domain dimensions $[0, 0, 0] \times [10, 10, 10]$ mm. For a spherical droplet, the curvature is uniform throughout the interface, denoted as $\kappa_{\Sigma} = \frac{2}{R} = 2000 \text{ m}^{-1}$. This curvature value is prescribed and held constant in simulations to avoid testing errors arising from curvature approximation and only verify the force-balanced non-orthogonality correction. Two error norms are evaluated to highlight the accuracy of the algorithm, namely

$$\begin{aligned}
 L_{\infty}(|\mathbf{v}|) &= \max(|\mathbf{v}_n - \mathbf{v}_e|) \\
 L_{\infty}(|\Delta p|) &= \frac{|\max(\Delta p_n) - \Delta p_e|}{\Delta p_e} \\
 \max(\Delta p_n) &= \max(p)_n - \min(p)_n
 \end{aligned} \tag{12.1}$$

Mesh type	Resolution $\frac{\Delta x}{L}$	max. non-ortho.		ResNonOrthCorr			FixNonOrthCorr($N_{non} = 1$)		
		global(θ_f) (°)	local(θ_f) (°)	$L_\infty(\mathbf{v})$ (m/s)	$L_\infty(\Delta p)$ (Pa)	CPU time (s)	$L_\infty(\mathbf{v})$ (m/s)	$L_\infty(\Delta p)$ (Pa)	CPU time (s)
blockMesh	1/30	0	0	2.4567e-11	1.4457e-14	432.32	9.1924e-11	8.2053e-15	384.54
	1/60	0	0	1.7332e-11	1.7583e-14	3138.92	3.0472e-10	2.8914e-14	3032.42
	1/90	0	0	7.9776e-12	2.3835e-14	8990.49	6.0617e-12	2.2076e-14	8633.76
perturbMesh	1/30	12.79	10.09	4.0199e-10	3.7432e-13	463.24	2.1683e-07	3.9894e-10	560.91
	1/60	14.45	10.87	2.8729e-10	3.6006e-13	3294.92	1.5253e-06	4.063e-09	5210.09
	1/90	13.54	11.00	1.7006e-09	1.8767e-12	9432.08	1.8398e-06	5.8021e-09	18328.48
polyMesh	1/30	30.81	1.48e-06	3.9613e-11	4.6731e-13	1653.41	4.3341e-11	1.6684e-13	1600.45
	1/60	33.42	1.57e-06	7.7984e-12	3.5556e-14	10256.68	9.0486e-12	1.2015e-13	9630.83
	1/90	30.81	1.71e-06	7.3637e-12	1.6743e-13	38833.63	1.9106e-12	1.1917e-14	37077.32

Table 9: Maximum global and near-interface non-orthogonality, the velocity and pressure jump errors, and CPU times for a stationary droplet in equilibrium at the end time $t_{end} = 0.1s$

where the subscript n and e denote the numerical and exact solutions correspondingly. The exact solutions are

$$\mathbf{v}_e = \mathbf{0} \text{ m/s,}$$

$$\Delta p_e = \sigma \kappa_\Sigma = 145.48 \text{ Pa.}$$

Table 9 presents CPU time, the final velocity and pressure jump errors for both control algorithms on various meshes, along with information regarding non-orthogonality. The non-orthogonality is corrected once per inner loop for FixNonOrthCorr in table 9, denoted as $N_{non} = 1$. The non-orthogonality at a cell face described in table 9 is quantified by the angle of intersection of the line connecting two adjacent cell centers and the face normal, i.e., θ_f depicted in fig. 41. Two maximum non-orthogonality are documented in table 9. The global maximum non-orthogonality represents the largest θ_f among cell faces throughout the computational domain Ω . Additionally, particular attention is given to the maximum non-orthogonality in the vicinity of the interface that actually causes force-imbalance resulting in parasitic currents. Specifically, the maximum non-orthogonality over the faces in the interface cell-layer and two adjacent cell layers is assessed.

When the pressure Poisson equation eq. (11.4) is solved on orthogonal meshes (i.e., blockMesh or orthogonal polyMesh), where the non-orthogonality is zero or near-zero, both

ResNonOrthCorr and FixNonOrthCorr perform similarly. Very minor disparities in two error norms and CPU time are observed, demonstrating a very low computational overhead of ResNonOrthCorr stopping criterion. On the orthogonal mesh, the explicit non-orthogonal contribution $(\cdot)_f^{\parallel}$ in eq. (11.11) is zero, implying that the non-orthogonal correction does not introduce new errors. The errors reported in table 9 for cases employing blockMesh arise solely from the orthogonal contribution. Notably, the errors approach the prescribed error tolerance, reaffirming the force-balance property inherent in the consistent discretization of pressure and body force gradients at cell faces [112], for the unstructured Finite Volume method [141].

The non-orthogonality of perturbMesh in table 9 is uniformly distributed throughout the computational domain, with both global and local non-orthogonality ranging between 10° and 15° - still very small and generally acceptable magnitudes. Even for acceptably small non-orthogonality, strong differences arise between ResNonOrthCorr and FixNonOrthCorr. The final $L_\infty(|\mathbf{v}|)$ and $L_\infty(|\Delta p|)$ values computed with ResNonOrthCorr are three to four orders of magnitude smaller than those obtained with FixNonOrthCorr. Furthermore, the CPU times for all three cases with varying resolutions employing ResNonOrthCorr are markedly lower than those using FixNonOrthCorr, especially with a reduction in approximately half the CPU time for the two higher-resolution cases. The force-balance impacts computational efficiency: a force-balanced discretization will recover the accurate steady state, causing less work for the solution algorithm and the linear solver, compared to the lack of force balance. Not ensuring force balance incurs acceleration of the fluid, whose velocity changes in time (until possibly reaching steady state), and should be divergence-free for incompressible fluid, which means more work for the solution algorithm and the pressure Poisson equation.

To thoroughly investigate the impact of the non-orthogonality correction number N_{non} on the errors and CPU time for the FixNonOrthCorr algorithm, additional tests are conducted with two distinct groups of cases featuring identical perturbMesh setups as outlined in table 9 but with larger values for the number of non-orthogonality corrections, i.e., $N_{non} = [2, 10]$, and present the details in table 10.

As N_{non} increases to 2, both final errors are reduced by two orders of magnitude. However, these results are still significantly outperformed by ResNonOrthCorr, whose CPU times are also significantly shorter.

When the non-orthogonal contribution is corrected 10 times in each inner loop, i.e., $N_{non} = 10$, for FixNonOrthCorr, the final errors exhibit minor differences compared to ResNonOrthCorr and approach the prescribed tolerance, suggesting that setting $N_{non} = 10$ for this static droplet test is sufficient to mitigate errors attributable to non-orthogonality to a satisfactory level. However, the CPU time required to achieve a similar level of accuracy with FixNonOrthCorr and $N_{non} = 10$ is approximately 70% higher than that with ResNonOrthCorr for each resolution. Figure 44 illustrates the temporal evolution and average non-orthogonal correction times. After sufficient corrections at the first time step, the non-orthogonality is corrected only around 5 times for ResNonOrthCorr with all resolutions, as shown in fig. 44a. Owing to the prescribed $N_{outer} = 4$, $N_{inner} = 1$ and $N_{non} = 10$, the correction times for FixNonOrthCorr are fixed accordingly, i.e., $N_{outer} \times N_{inner} \times N_{non} = 40$, which explains the horizontal blue dashed line of FixNonOrthCorr in fig. 44a. Figure 44b presents the average non-orthogonal correction times per time step regarding fig. 44a. The correction times rise slightly as the resolution increases for ResNonOrthCorr, whereas the correction times are unchanged for FixNonOrthCorr regardless of the resolution. The fig. 45 visually depicts the temporal evolution of $L_\infty(|\mathbf{v}|)$ with different control methods and indicates a trend: the errors from FixNonOrthCorr decreases as N_{non} rises in the context of the non-orthogonal perturbMesh. When the specified N_{non} is sufficiently large, such as $N_{non} = 10$ in this case, FixNonOrthCorr works same as ResNonOrthCorr, as evidenced by the complete overlap of the blue stars from ResNonOrthCorr with the red transparent stars from FixNonOrthCorr with $N_{non} = 10$ in fig. 45. Finally, fig. 46 visually represents the final velocity field on perturbMesh with ResNonOrthCorr and FixNonOrthCorr, where the orientation and length of each glyph arrow signify the direction and magnitude scaled by a factor of $1e-7$ of velocity at a cell center.

In addition to the prescribed Green Gauss method, a least-square method named *point-CellsLeastSquares* were explored. This method utilizes point-neighbor cells as the stencil to calculate the gradient, allowing us to investigate the impact of different cell center gradient reconstruction methods on non-orthogonality correction. As depicted in table 11, the errors obtained using the higher-order accurate least-square method are reduced at the final time for both ResNonOrthCorr and FixNonOrthCorr ($N_{non} = 10$). However, the more complex calculation involved in the least-square gradient results in higher computational costs, as evidenced by the increased CPU time shown in table 11. The temporal evolution of velocity

errors for different gradient reconstructions is illustrated in fig. 55.

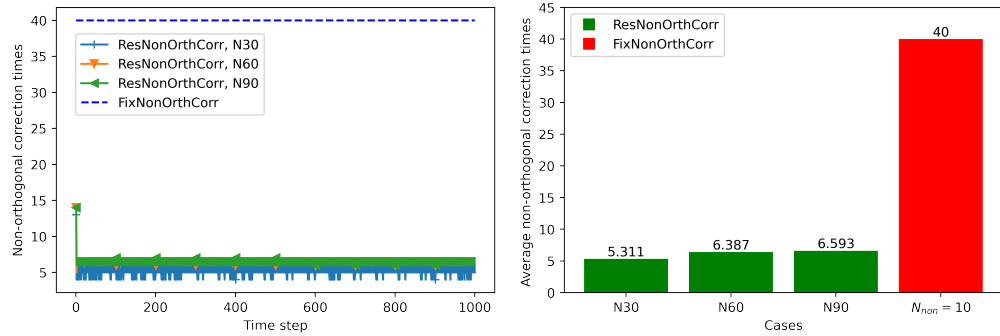
Table 12 presents the final results of errors and CPU time on perturbMesh using the MULES, another Volume of Fluid (VoF)-based interface capturing method in OpenFOAM® [158]. In contrast to the geometric isoAdvector [159] utilized earlier, MULES is an algebraic VoF solver that employs the Flux Corrected Transport (FCT) technique [160, 161]. The ResNonOrthCorr is combined with MULES. As shown in table 12, the errors obtained from MULES + ResNonOrthCorr approach the preset tolerance. On the other hand, the CPU time for all resolutions is very similar to the preceding results obtained from isoAdvector + ResNonOrthCorr. Our method demonstrates excellent compatibility with different phase advection methods. Figure 56 illustrates the velocity errors over time for MULES + ResNonOrthCorr. The combination of ResNonOrthCorr with MULES yields even better results for this stationary droplet case.

Mesh type	Resolution $\frac{\Delta x}{L}$	FixNonOrthCorr($N_{non} = 2$)			FixNonOrthCorr($N_{non} = 10$)		
		$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)	$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)
perturbMesh	1/30	3.81861e-09	8.7535e-12	527.45	4.0199e-10	3.3798e-13	857.35
	1/60	2.9294e-08	8.5609e-11	4947.63	2.8729e-10	1.3269e-12	5771.95
	1/90	1.1993e-07	2.6584e-10	16894.48	1.7006e-9	2.2078e-12	15530.31

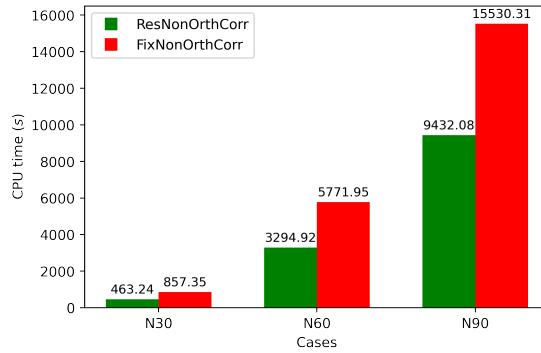
Table 10: The performances of FixNonOrthCorr with non-orthogonality loop numbers larger than one for a stationary droplet in equilibrium case on perturbMesh.

Mesh type	Resolution $\frac{\Delta x}{L}$	ResNonOrthCorr			FixNonOrthCorr($N_{non} = 10$)		
		$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)	$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)
perturbMesh	1/30	6.3041e-11	1.4322e-12	663.04	6.3041e-11	1.4322e-12	1066.53
	1/60	6.5157e-11	7.1151e-13	5166.65	6.5157e-10	6.8416e-13	7171.11
	1/90	4.0325e-11	5.7556e-12	13342.54	4.0325e-11	5.7281e-12	19408.88

Table 11: The performances of ResNonOrthCorr and FixNonOrthCorr($N_{non} = 10$) with least-square cell center gradient reconstruction method for a stationary droplet in equilibrium case on perturbMesh.



(a) perturbMesh: temporal evolution of the number of non-orthogonality corrections. (b) perturbMesh: average number of non-orthogonality corrections.



(c) perturbMesh: CPU time; average speedup of ResNonOrthCorr 174.97%.

Figure 44: The temporal evolution and average of non-orthogonal correction times, and the CPU time with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = 10$) for a stationary droplet in equilibrium on perturbMesh with different resolutions.

In contrast to perturbMesh, the distribution of non-orthogonality within the polyMesh is irregular. As illustrated in table 9, the maximum local θ_f values are notably small, while the maximum global θ_f values exceed 30° for all resolutions. This observation indicates that cells near the interface exhibit nearly orthogonal characteristics. Consequently, despite only using a single correction for non-orthogonality (i.e., once in each inner loop, $N_{non} = 1$), the velocity and pressure jump outcomes from FixNonOrthCorr on polyMesh align with the high

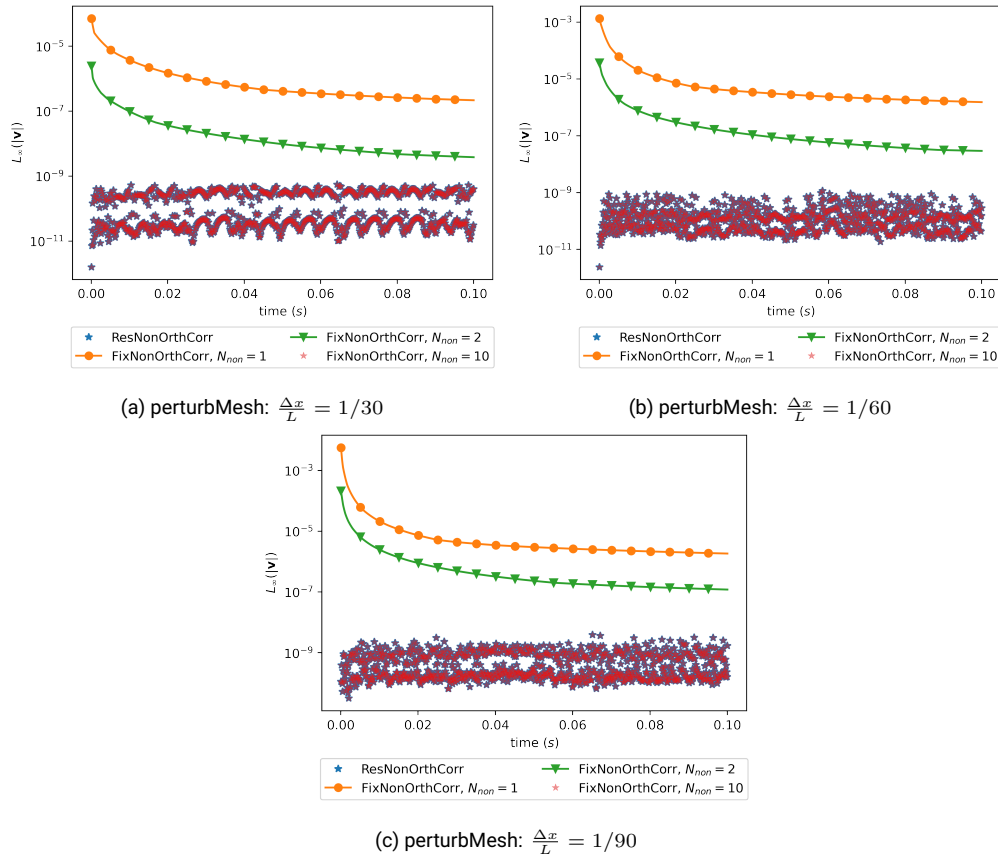


Figure 45: The temporal evolution of velocity error norm $L_{\infty}(|\mathbf{v}|)$ with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = [1, 2, 10]$) for a stationary droplet in equilibrium on perturbMesh with different resolutions.

accuracy achieved on blockMesh. The substantial non-orthogonality does not necessitate a higher N_{non} , rendering the determination of an appropriate N_{non} more arbitrary and intricate for users. ResNonOrthCorr, in contrast, circumvents this issue and yields satisfactory results.

To conclude the results for the stationary droplet, the ResNonOrthCorr algorithm successfully ensures force-balance for the surface tension force and the pressure gradient on non-orthogonal unstructured finite volume discretization. It adjusts the number of required

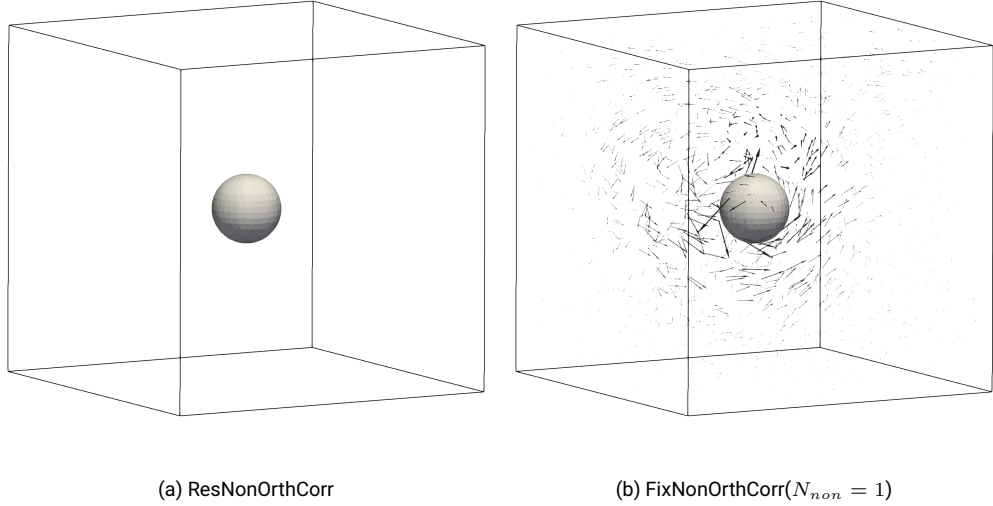


Figure 46: The velocity field of the stationary droplet in equilibrium on perturbMesh with the resolution $\frac{\Delta x}{L} = 1/60$ at $t = t_{end}$: the glyph arrows are scaled by $1e-7m/s$.

Mesh type	Resolution $\frac{\Delta x}{L}$	ResNonOrthCorr: MULES		
		$L_{\infty}(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)
perturbMesh	1/30	6.1017e-12	5.6519e-13	371.47
	1/60	9.6775e-11	1.2905e-11	3398.93
	1/90	1.8768e-11	1.2631e-11	10096.52

Table 12: The performances of combining MULES phase advection algorithm with ResNonOrthCorr for a stationary droplet in equilibrium case on perturbMesh.

non-orthogonality iterations to satisfy eq. (11.15), a very cost-effective stopping criterion. The total CPU time is significantly reduced compared to the heuristic approach, since the number of non-orthogonal corrections is kept to a minimum. Finally, and equally impactful, ResNonOrthCorr removes the number of non-orthogonality corrections as a heuristic user-defined parameter from the CFD simulation. It is found in this verification study that $N_{non} = 10$ ensures sufficient accuracy (at a much higher computational cost); however,

finding this number for a highly resolved geometrically complex 3D microfluidic simulation is nearly impossible, as it is not possible to know the maximal interface-local non-orthogonality of all the cells that will be visited by the fluid interface during a simulation.

12.2 Stationary water column in equilibrium

In order to investigate the balance between gravitational force and the associated pressure gradient, a stationary water column in equilibrium is under consideration. In this analysis, the effects of surface tension force and viscosity are neglected. The test employs the same discretization scheme for density and pressure gradients to satisfy the fundamental force-balance requirement. It is crucial to note that inappropriately estimating \mathbf{x} in $(\mathbf{g} \cdot \mathbf{x})$ can also deteriorate the force-balance [162–165], particularly for the multiphase flows characterized by high density-ratios. This is because any error from the estimation of \mathbf{x} is amplified by $(\rho^- - \rho^+)|\mathbf{g}|$, which approaches 10^4 for water/air on earth. However, addressing the accuracy improvement of \mathbf{x} estimation falls outside the scope of this study. Interested readers are directed to [162, 165] for further details. For an inviscid water column in equilibrium, the interface Σ between water and air is flat, resulting in a constant value for $(\mathbf{g} \cdot \mathbf{x}_\Sigma)$.

The cubic container of the water column is used as the computation domain Ω , with dimensions of $[0, 0, 0] \times [1, 1, 1]$ m. The water occupies the region where $z_\Sigma \leq 0.5145$ m within Ω . All the boundaries are treated as walls, except for the top boundary, which is modeled by the open-air boundary condition. The relevant properties of water and air, as well as gravity, are defined in table 8. The error norms from eq. (12.1) are employed, along with the exact solution

$$\begin{aligned}\mathbf{v}_e &= \mathbf{0} \text{ m/s} \\ \Delta p_e &= (\rho^- - \rho^+)|\mathbf{g}|z_\Sigma = 5040.95633874 \text{ Pa.}\end{aligned}$$

Similar to the stationary droplet case, when non-orthogonality is uniformly zero across the entire computational domain for `blockMesh`, `ResNonOrthCorr` and `FixNonOrthCorr` ($N_{non} = 1$) exhibit closely comparable performance in terms of velocity and pressure jump errors, as well as CPU time, presented in table 13.

The maximal local non-orthogonality exceeds 10° for `perturbMesh`, leading to noticeable disparities in errors between `ResNonOrthCorr` and `FixNonOrthCorr`. Specifically, with

Mesh type	Resolution $\frac{\Delta x}{L}$	max. non-ortho.		ResNonOrthCorr			FixNonOrthCorr($N_{non} = 1$)		
		global(θ_f) (°)	local(θ_f) (°)	$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)	$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)
blockMesh	1/30	0	0	6.1977e-14	7.1573e-13	686.11	1.4301e-13	7.1898e-13	675.24
	1/60	0	0	7.8625e-12	4.0783e-13	4052.7	7.8600e-12	4.0801e-13	4018.51
	1/90	0	0	6.6016e-12	1.7352e-13	10641.22	6.6131e-12	1.7316e-13	10587.81
perturbMesh	1/30	12.79	12.16	1.3731e-10	6.0642e-13	725.66	3.5756e-05	2.7689e-11	725.8
	1/60	14.45	13.64	4.9291e-11	3.3008e-13	4154.27	1.0977e-04	9.9943e-11	4349.29
	1/90	13.54	12.77	1.7483e-11	5.9791e-12	10872.64	6.7688e-05	3.0113e-11	11505.59
polyMesh	1/30	60.41	58.88	1.4677e-09	1.8047e-08	3232.11	7.5769e-03	8.5541e-08	3516.54
	1/60	63.30	58.98	1.1224e-08	2.4435e-09	16538.47	3.3555e-02	2.7190e-07	21090.88
	1/90	60.64	58.73	4.5175e-08	2.8148e-08	49231.96	8.2165e-03	6.1529e-08	57566.64

Table 13: The maximum non-orthogonalities in the global region and the local region near the interface, the velocity, pressure jump errors, and CPU time for a stationary water column in equilibrium at the end time $t_{end} = 0.1s$

regard to the velocity field, errors with ResNonOrthCorr are 5 to 7 orders of magnitude smaller than those observed with FixNonOrthCorr. Although the CPU times for simulations with ResNonOrthCorr control are slightly lower than those using FixNonOrthCorr, the error discrepancies are significant.

The polyMesh for this test shows the highest local non-orthogonality when compared with blockMesh and perturbMesh. Although the mesh generation is the same as for table 9, here the fluid interface touches the domain boundary, that has maximal non-orthogonality. Despite the very large non-orthogonality of $\approx 60^\circ$, ResNonOrthCorr effectively achieves force balance with velocity error magnitudes of $\approx 1e-8$. In contrast, parasitic velocities arising from the use of FixNonOrthCorr reach the magnitude of $1e-2$. The CPU times of ResNonOrthCorr are additionally comparatively lower than those of FixNonOrthCorr.

Table 14 contains the errors and CPU times from FixNonOrthCorr with $N_{non} > 1$ on both perturbMesh and polyMesh. Clearly, the errors decrease with increasing N_{non} , but significantly more CPU time is required as N_{non} increases for each resolution on both meshes. Figure 57 and Figure 58 illustrate the temporal evolution of the velocity error norm $L_\infty(|\mathbf{v}|)$ with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = [1, 2, 10]$) for a stationary water column in equilibrium on perturbMesh and polyMesh respectively. After correcting the explicit

Mesh type	Resolution $\frac{\Delta x}{L}$	FixNonOrthCorr($N_{non} = 2$)			FixNonOrthCorr($N_{non} = 10$)		
		$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)	$L_\infty(\mathbf{v})$ (m/s)	$L(\Delta p)$ (Pa)	CPU time (s)
perturbMesh	1/30	8.0268e-10	1.8642e-11	748.68	1.3731e-10	1.9503e-10	965.61
	1/60	1.5662e-09	1.4149e-10	4321.92	4.9291e-11	1.1832e-10	5989.96
	1/90	1.1303e-09	1.4348e-10	11422.03	1.7483e-11	1.6857e-09	16392.02
polyMesh	1/30	1.8107e-04	2.2694e-09	3466.42	1.7267e-10	3.0471e-08	4447.46
	1/60	9.5487e-04	4.0979e-09	18902.18	1.0068e-08	1.6252e-09	24490.22
	1/90	7.5324e-05	5.4888e-10	54671.07	1.1127e-08	8.1320e-09	76854.99

Table 14: The performances of FixNonOrthCorr with non-orthogonality loop numbers larger than one for a stationary water column in equilibrium case on perturbMesh and polyMesh.

non-orthogonal part of the face gradient a sufficient number of times, e.g., $N_{non} = 10$ for perturbMesh, the final errors in velocity converge to the results obtained with ResNonOrthCorr as depicted in fig. 57. For the polyMesh owning the largest non-orthogonality, for $N_{non} = 10$, FixNonOrthCorr still cannot recover force balance as accurately as ResNonOrthCorr do, as illustrated in fig. 58, where the red star symbols from FixNonOrthCorr with $N_{non} = 10$ are still slightly higher than the blue stars from ResNonOrthCorr. The significantly increased CPU time associated with the higher N_{non} , as shown in tables 13 and 14, implies the poor balance between the result accuracy and the computational costs for FixNonOrthCorr method. On the contrary, ResNonOrthCorr can achieve high accuracy with relatively much lower computational costs. Figure 47 shows the final velocity field using ResNonOrthCorr and FixNonOrthCorr on perturbMesh with the middle mesh resolution. The parasitic currents are distributed randomly above the flat interface in the results using FixNonOrthCorr, whereas they cannot be observed at the same scale when adopting ResNonOrthCorr.

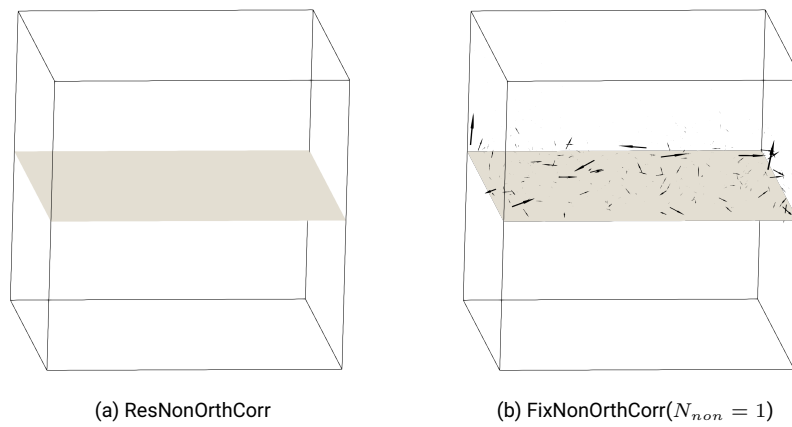


Figure 47: The velocity field of the stationary water column in equilibrium on perturbMesh with the resolution $\frac{\Delta x}{L} = 1/60$ at $t = t_{end}$: the glyph arrows are scaled by $1e-5m/s$.

Summary and outlook

Within this thesis, a method ensuring numerical consistency of the single-field incompressible two-phase momentum convection, discretized by the unstructured collocated Finite Volume Method, is proposed to deal with challenging high density ratios problems. The method can be seamlessly integrated into any two-phase flow simulation method relying on the collocated FVM for equation discretization of two-phase single-field Navier-Stokes equations. This integration involves incorporating a geometrical computation of area fractions from the approximated fluid interface and an auxiliary density equation to the solution algorithm. An analysis connecting mass conservation, phase indicator function conservation, and momentum convection is provided, theoretically justifying the necessity for the numerical consistency of the cell-centered density computed by a mass flux identical to the one used in the two-phase momentum convective term.

This consistent method is first implemented on an existing Level Set / Front Tracking method [2, 35, 85], where the numerical instability is lost in simulating two-phase flows with high density ratios. The new approach is termed ρ LENT. Results confirm the method's accuracy and stability, with exact recovery of canonical droplet translation and successful simulation of droplets with sub-millimeter diameters. Validation against experiments demonstrates accurate capture of phenomena such as the oscillation frequency of ellipsoidal droplets and the deformation of rising bubbles, affirming the method's reliability in practical applications.

While substantial stability and accuracy improvements have been achieved for the ρ LENT method when applied to high density-ratio flows, further work is required to make predictive simulations for technical applications feasible. The current iso-surface reconstruction approach limits the method to cases with small to moderate interface deformation. More efforts are still needed for this method to simulate cases containing more complex processes such as

breakup and coalescence, where structures with different scales are needed to be captured and resolved, which usually needs more computational resources. Therefore, the capability of parallel computation is indispensable for ρ LENT in the future development. The ρ LENT, which is implemented on OpenFOAM[®] platform, relies on OpenFOAM[®]'s Message Passing Interface (MPI) parallelization. The unstructured VOF method requires only two cell-layers adjacent to the boundary shared between two MPI processes to discretize PDE. A naive MPI-parallelization of the ρ LENT algorithm requires the Front to be communicated across the MPI boundary. However, it is found that this approach scales poorly and complicates the parallel implementation beyond the point of utility. Additional investigation will be conducted on higher-order compact interpolation (e.g. Radial-Basis-Function interpolation) or approximation (e.g. quadratic surface approximation) to reduce reconstruction errors when interpolating signed distances. The second-order interpolation is used to find the root points of the iso-surface - the points that define Front - without actually performing the polygonization of the surface mesh. Sets of root-points thus recovered for each cell have a centroid associated with them. This inaccurate (mean-value) dual-contouring centroid is then projected onto the higher-order interpolation (approximation) of the fluid interface - significantly reducing such as volume conservation errors. Dual contouring then proceeds to polygonize the surface (Front) using higher-order centroids. Another point of future work involves the proper handling of viscous term in Navier-Stokes equations. Similar with density ratio, high viscosity ratio can also result in numerical instability. An extended discussion about viscous term modeling are given in appendix 4.

Solving the auxiliary density advection equation looks unnecessary at first glance considering the definition of mixed density from volume fraction, when transplanting the consistent method to a flux-based VOF method. However, an analyse in section 6.4 shows that the equivalence between conservative mass equation and volume fraction advection equation establishes strictly in mathematics. On the discrete level, the analysis reveals that achieving equivalence necessitates specific conditions: first-order Euler temporal discretization for momentum conservation, absence of flux limiting, and utilization of first-order quadrature for integrating the fluxed phase-specific volume. Any deviation from the conditions results in errors proportional to density differences, potentially leading to significant distortions in interface shape as well as catastrophic failure. To verify these points, the consistent method

is extended to an unstructured geometric VOF method, i.e., plicRDF-isoAdvector [6, 36, 49]. This adaptation requires straightforward geometric calculations of upwind area fractions from existing geometric VOF interface approximations. The demonstration of equivalence on the discrete level between mass flux scaled from phase-specific volume and the solution of an auxiliary density equation is corroborated through verification with challenging inviscid translating droplet cases and validation against experiments.

Section 6.4 reveals the limitation of the temporal and spatial discretization scheme when computing the mass flux by scaling the fluxed phase-specific volume over a time step. The explicit integration of the volume fraction is problematic for approximating the mass flux at different time steps t^k , if the mass flux is approximated by scaling the phase-specific fluxed volume, e.g., as $\frac{(\rho^- - \rho^+) |V_f^\alpha|_s}{\Delta t}$, because by scaling, a mean value of the mass flux over the time step Δt is calculated. In section 3.5.4, there is no phase-specific volume $|V_f^\alpha|$ in the Level Set / Front Tracking method that could be scaled, so $\rho_f F_f$ is recovered geometrically as $\rho_f F_f \approx [(\rho^- - \rho^+) \alpha_f + \rho^+] F_f$, with α_f geometrically reconstructed from signed distances. This can be done at any point in time t , given the affordable way of obtaining $\Sigma(t)$, as the Front can be moved with higher-order accuracy by moving its points using a higher-order explicit method that utilizes existing velocities. However, even for ρ LENT, Crank-Nicolson fails with $\rho_f F_f \approx [(\rho^- - \rho^+) \alpha_f + \rho^+] F_f$. The same scheme is tried for the unstructured VOF, geometric (isoAdvector) and algebraic (interFoam/MULES/FCT) and it also failed. In the future work, the fractional step method will be attempted, which might allow to geometrically consistently advect momentum by re-using directly the VOF geometrical advection and thus retain its CFL condition for the momentum convection term. The topic of higher-order integration in this context is not straightforward for unstructured methods, especially since it is not expected to integrate the momentum equation explicitly entirely. Higher-order temporal integration will also be tracked in the future work.

One more interesting point deserved to further explore in the future is how to adapt the geometric interface tracking/capturing methods to meet the geometric constraints according to the boundary's type. An example is the adjustment of plicRDF on the cyclic boundary, as addressed in appendix 3. In the cyclic boundary condition, the pairing cyclic boundaries should be treated as a single internal patch, which constrains the calculation of the signed distances of the cells near the cyclic boundaries and further, the interface center and normal

and the curvature. Incorrect estimation of the signed distances at cyclic boundaries results in catastrophic failure in simulation, as shown in appendix 3. It is notable that in addition to cyclic boundary condition there are many other geometric boundary conditions, like symmetry condition, axis-symmetric (wedge) condition and processor condition. The plicRDF should be adapted for each according to its special geometric constraints. Actually, the geometric boundaries effect not only the plicRDF but also all interface tracking/capturing methods, which need reconstruct the interface and make use of the geometric information of interface, such as the FTM, LSM or the hybrid methods.

A novel approach is proposed in section 9.6 to address non-orthogonality issues in the unstructured Finite Volume method, offering high precision, determinism, and computational efficiency. This proposed method achieves a delicate equilibrium among pressure gradient, gravity force, and surface tension force at fluid interfaces by leveraging the residual norm of the solver, ensuring precise force balance without imposing substantial computational overhead. Notably, even on intricately perturbed hexahedral meshes where artificial error cancellation is impractical, this method operates effectively. By integrating the linear solver's norm and tolerance into a deterministic stopping criterion, the optimal number of non-orthogonality corrections required for maximum accuracy is established, eliminating the need for algorithmic adjustments or trial-and-error-based correction selection. Furthermore, the results demonstrate a remarkable enhancement in computational efficiency, yielding an order-of-magnitude improvement compared to traditional heuristic approaches.

In section 3.5.4, an iterative and accuracy-driven algorithm SAAMPLE [2] is adopted to solve the discretized pressure-velocity system. In SAAMPLE, extra update criterion based on the convergence of volumetric flux F_f are added to the loop. A combination of the residual control method in section 9.6 with SAAMPLE will be investigated at a future time to improve whole accuracy and stability of a solver. One work that is left for the future exploration involves modeling the gravitational term $(\mathbf{g} \cdot \mathbf{x})\nabla\rho$. In section 9.6, the necessity of consistent discretization scheme for pressure gradient and density gradient $\nabla\rho$ is addressed. However, the estimation of $\mathbf{g} \cdot \mathbf{x}$ remains under-discussed. The estimation is nontrivial considering that its error is amplified by $\nabla\rho$, which is a large value in a interfacial cell filled with two phases with high density ratio. A preliminary discussion about the approximation of $\mathbf{g} \cdot \mathbf{x}$ is given in appendix 5.

Acknowledgments Funded by the German Research Foundation (DFG) – Project-ID 265191195 – SFB 1194. Calculations for this research were conducted on the Lichtenberg high performance computer of the TU Darmstadt.

The computations with DYJEAT codes were granted access to the HPC resources of CALMIP supercomputing center under the allocation 2023 P18043.

Appendices

1 Supplementary results

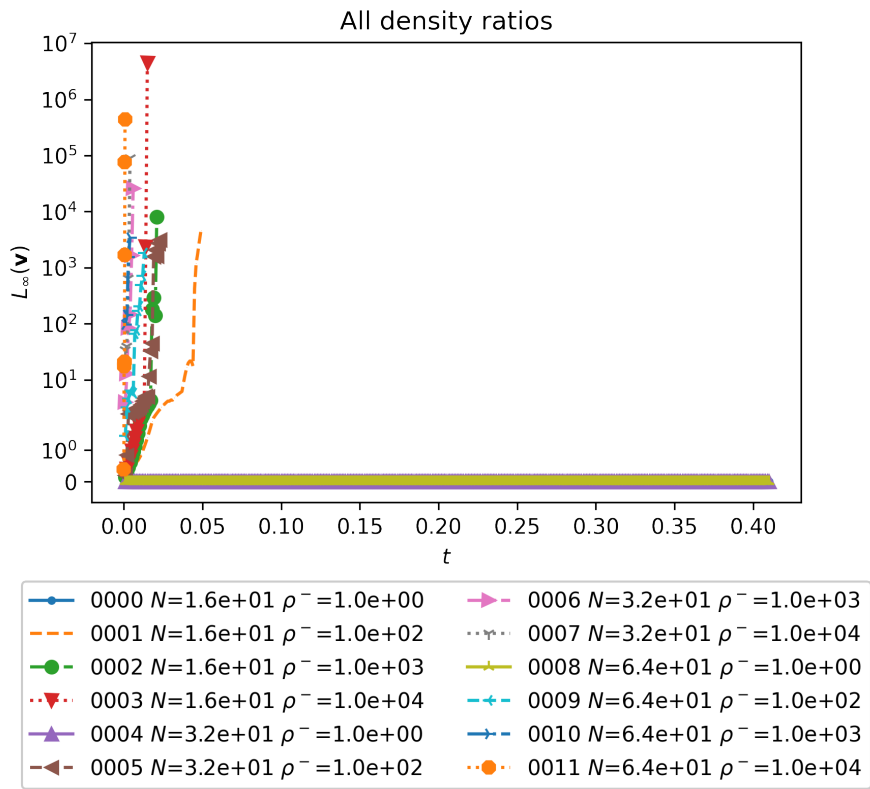


Figure 48: Full figure of fig. 16a

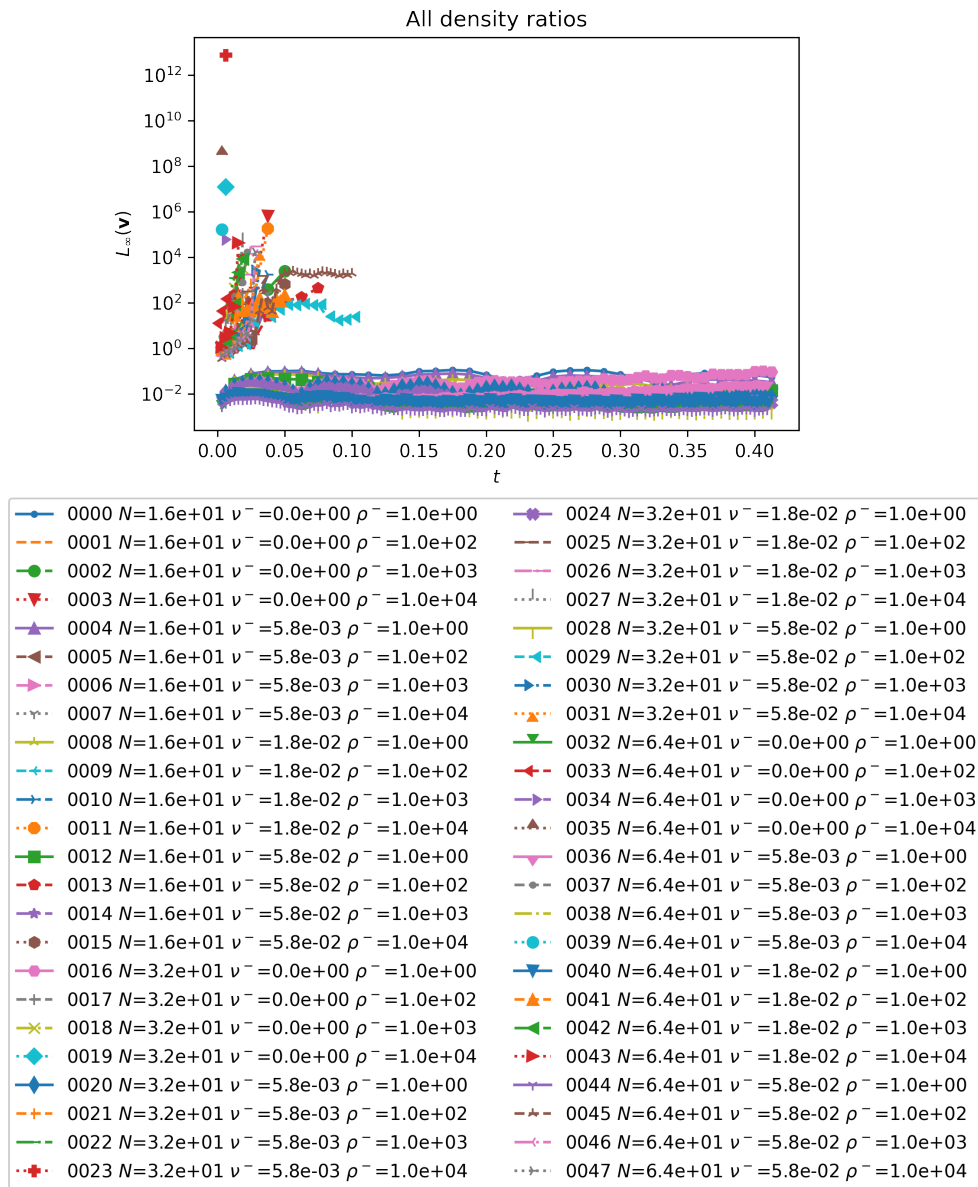
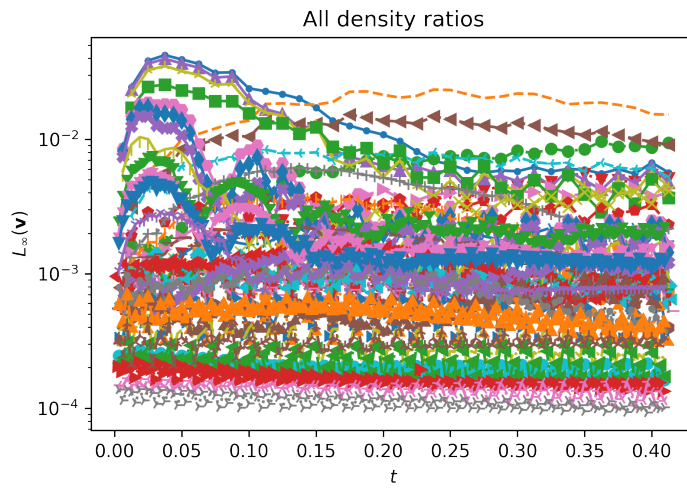


Figure 49: Full figure of fig. 18a



0000 $N=1.6e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+00$	0024 $N=3.2e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+00$
0001 $N=1.6e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+02$	0025 $N=3.2e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+02$
0002 $N=1.6e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+03$	0026 $N=3.2e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+03$
0003 $N=1.6e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+04$	0027 $N=3.2e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+04$
0004 $N=1.6e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+00$	0028 $N=3.2e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+00$
0005 $N=1.6e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+02$	0029 $N=3.2e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+02$
0006 $N=1.6e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+03$	0030 $N=3.2e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+03$
0007 $N=1.6e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+04$	0031 $N=3.2e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+04$
0008 $N=1.6e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+00$	0032 $N=6.4e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+00$
0009 $N=1.6e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+02$	0033 $N=6.4e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+02$
0010 $N=1.6e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+03$	0034 $N=6.4e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+03$
0011 $N=1.6e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+04$	0035 $N=6.4e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+04$
0012 $N=1.6e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+00$	0036 $N=6.4e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+00$
0013 $N=1.6e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+02$	0037 $N=6.4e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+02$
0014 $N=1.6e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+03$	0038 $N=6.4e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+03$
0015 $N=1.6e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+04$	0039 $N=6.4e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+04$
0016 $N=3.2e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+00$	0040 $N=6.4e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+00$
0017 $N=3.2e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+02$	0041 $N=6.4e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+02$
0018 $N=3.2e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+03$	0042 $N=6.4e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+03$
0019 $N=3.2e+01$ $v^- = 0.0e+00$ $\rho^- = 1.0e+04$	0043 $N=6.4e+01$ $v^- = 1.8e-02$ $\rho^- = 1.0e+04$
0020 $N=3.2e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+00$	0044 $N=6.4e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+00$
0021 $N=3.2e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+02$	0045 $N=6.4e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+02$
0022 $N=3.2e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+03$	0046 $N=6.4e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+03$
0023 $N=3.2e+01$ $v^- = 5.8e-03$ $\rho^- = 1.0e+04$	0047 $N=6.4e+01$ $v^- = 5.8e-02$ $\rho^- = 1.0e+04$

Figure 50: Full figure of fig. 18b

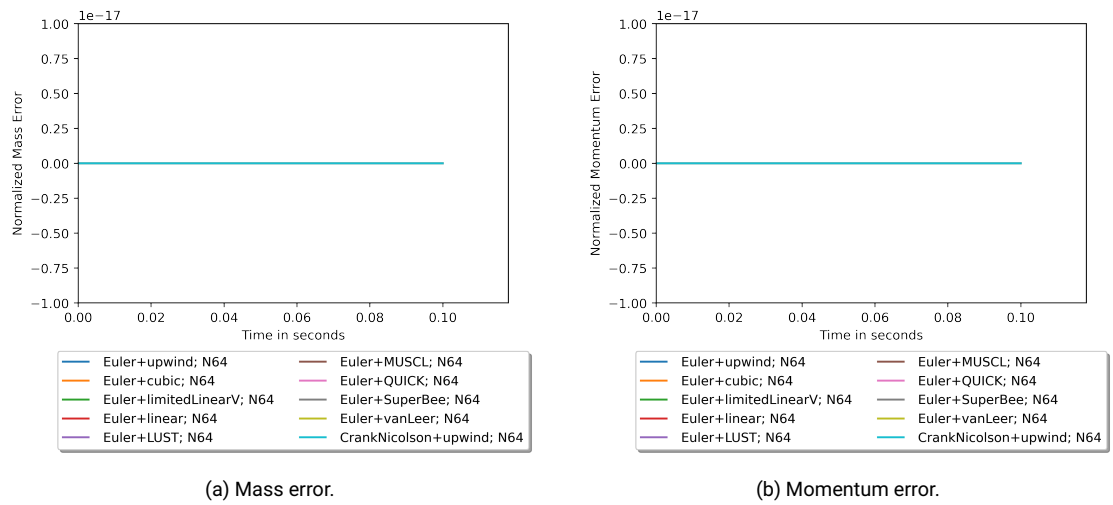


Figure 51: Temporal evolution of normalized mass and momentum conservation error with different schemes for the case of Translating droplet in ambient flow: interIsoFoam, $N = 64$, density ratio = 1.

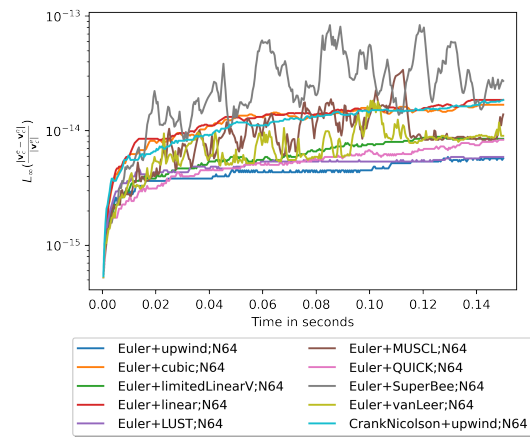


Figure 52: Temporal evolution of the velocity error norm $L_\infty(\mathbf{v})$ with pure advection - combining 10 schemes, density ratio is 1, mesh resolution is $N = 64$. All schemes are stable.

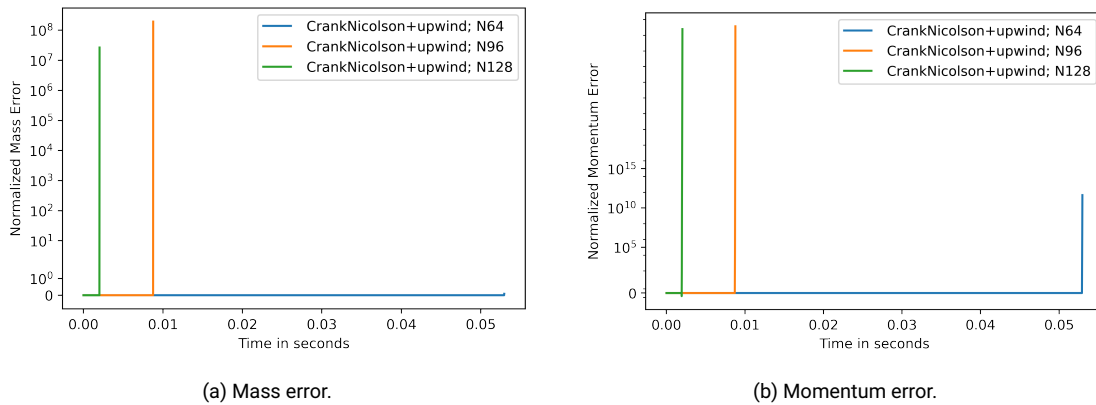


Figure 53: Temporal evolution of normalized mass and momentum conservation error using CrankNicolson + upwind with different resolutions for the case of Translating droplet in ambient flow: interIsoFoam, $N = 64, 96, 128$, density ratio = 10^6 .

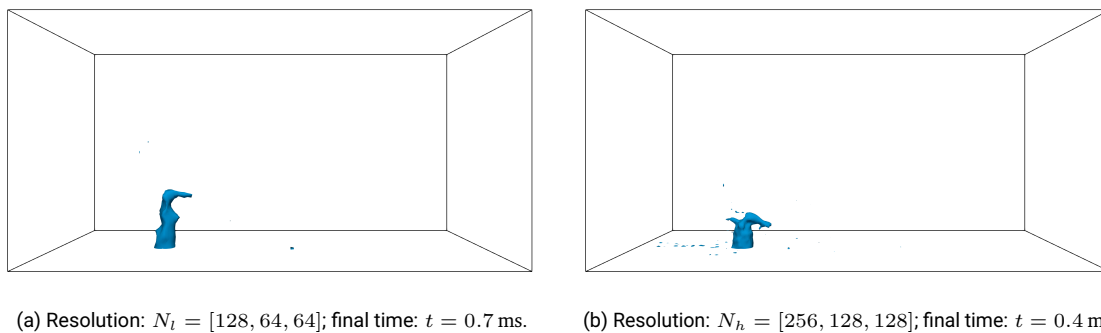


Figure 54: The shape of the exploded injected liquid with interIsoFoam (Euler and cubic, density ratio: 816, CFL number: $CFL = 0.2$).

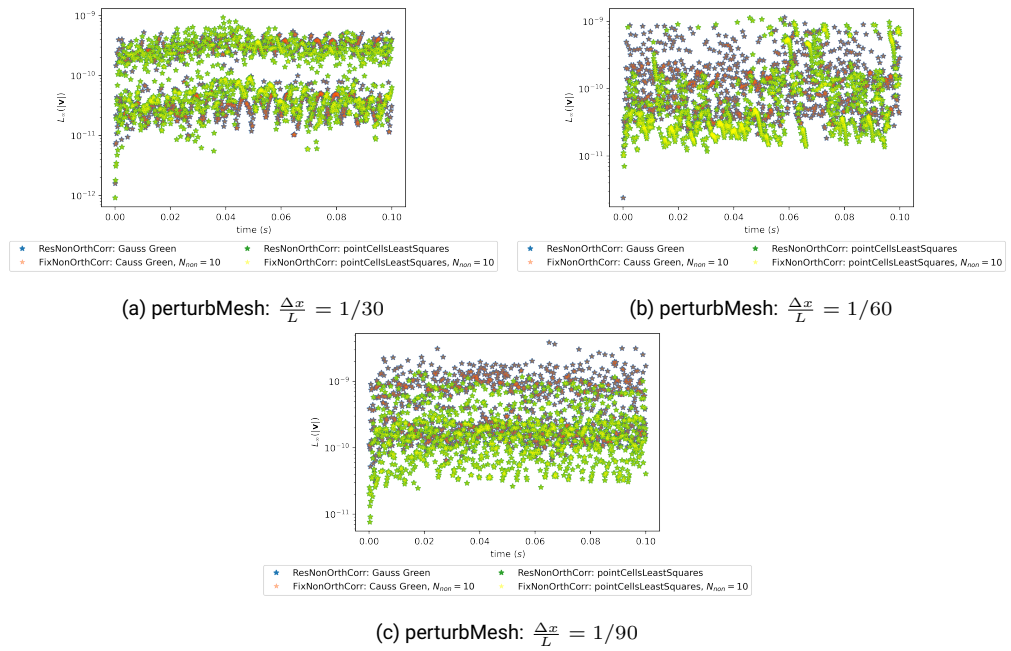
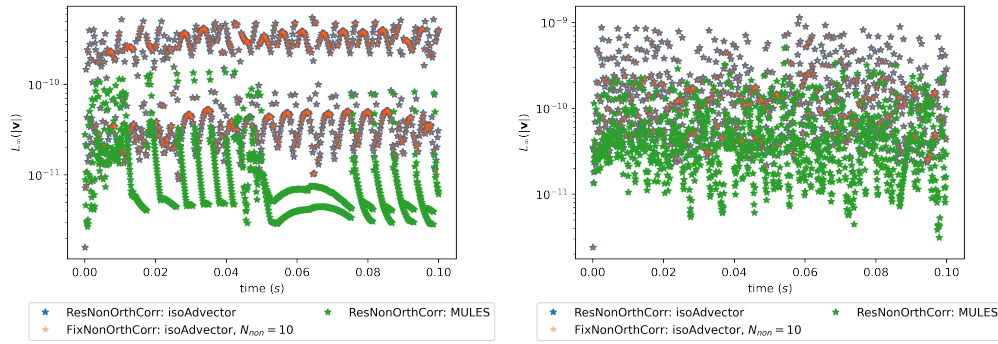
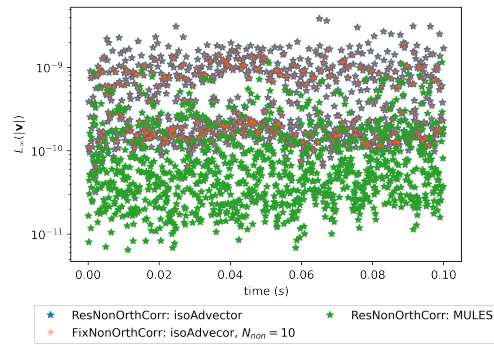


Figure 55: The temporal evolution of velocity error norm $L_\infty(|\mathbf{v}|)$ using ResNonOrthCorr and FixNonOrthCorr ($N_{non} = 10$) with least-square gradient reconstruction for a stationary droplet in equilibrium on perturbMesh with different resolutions. These results show that there is no influence on the gradient scheme on the proposed method.



(a) perturbMesh: $\frac{\Delta x}{L} = 1/30$

(b) perturbMesh: $\frac{\Delta x}{L} = 1/60$



(c) perturbMesh: $\frac{\Delta x}{L} = 1/90$

Figure 56: The temporal evolution of velocity error norm $L_\infty(|\mathbf{v}|)$ using MULES with ResNonOrthCorr for a stationary droplet in equilibrium on perturbMesh with different resolutions. These results show that the proposed method is directly applicable to the algebraic VOF method.

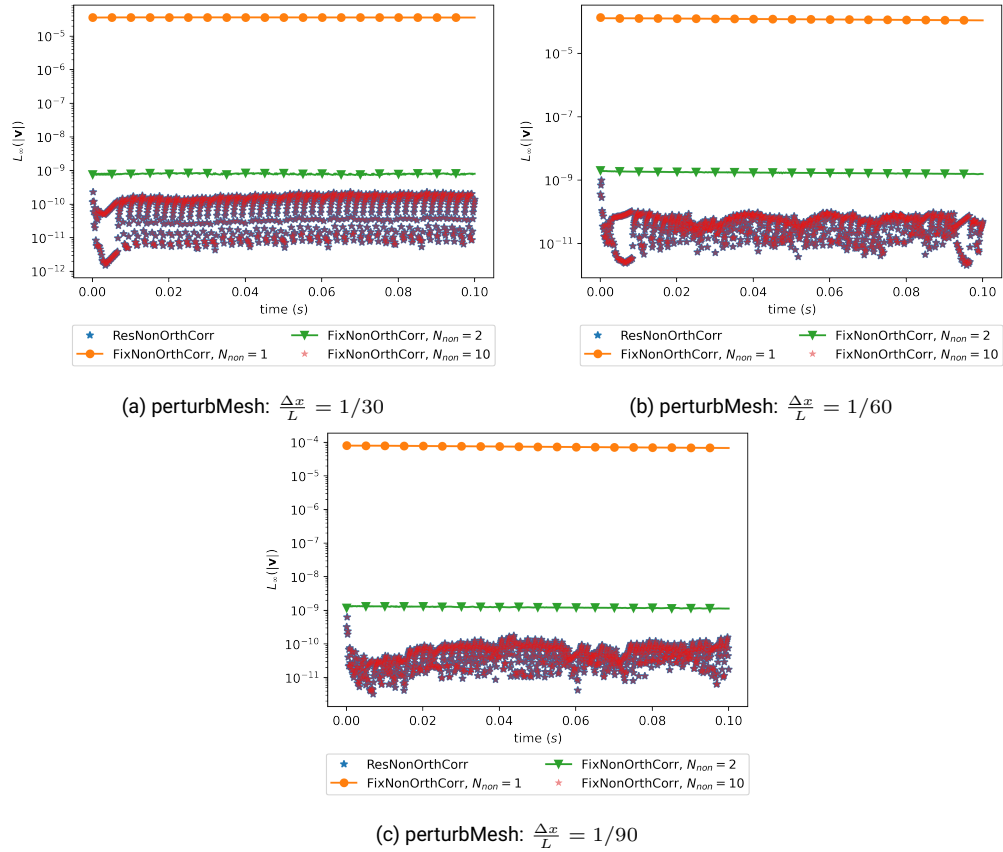
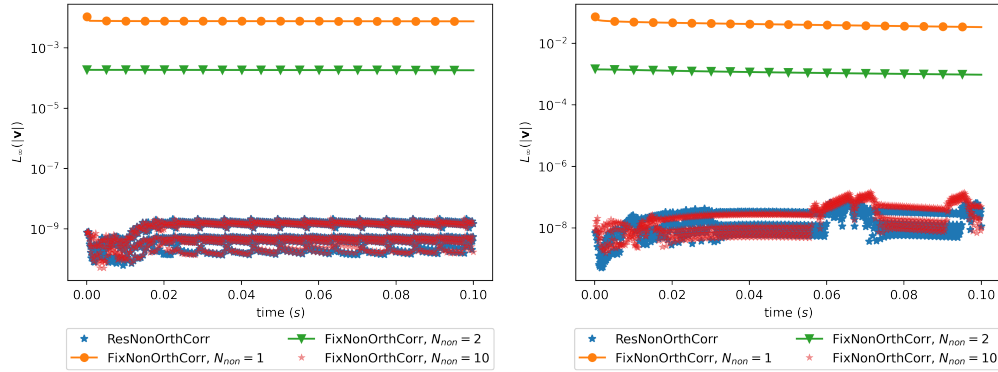
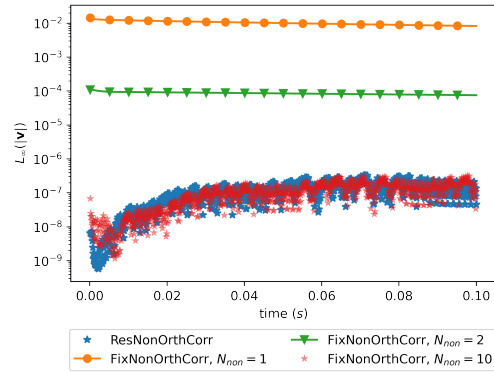


Figure 57: The temporal evolution of velocity error norm $L_\infty(|\mathbf{v}|)$ with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = [1, 2, 10]$) for a stationary water column in equilibrium on perturbMesh with different resolutions. Although $N_{non} = 10$ achieves force-balance, it is a problem-dependent "free" parameter that the proposed method does not use.



(a) polyMesh: $\frac{\Delta x}{L} = 1/30$

(b) polyMesh: $\frac{\Delta x}{L} = 1/60$



(c) polyMesh: $\frac{\Delta x}{L} = 1/90$

Figure 58: The temporal evolution of velocity error norm $L_\infty(|\mathbf{v}|)$ with ResNonOrthCorr and FixNonOrthCorr ($N_{non} = [1, 2, 10]$) for a stationary column in equilibrium on polyMesh with different resolutions. Although the polyhedral mesh has very low non-orthogonality in the bulk, near walls, even for a cubic domain, non-orthogonality is substantial, and the force-imbalance is efficiently and effectively restored by the proposed method.

2 Least square method

The description below is referred to Demirdžić and Muzaferija [150] and Jasak and Weller [166]. The basic linear spatial distribution of a dependent variable ϕ is adopted

$$\phi(\mathbf{x}) = \phi_P + (\nabla\phi)_P \cdot (\mathbf{x} - \mathbf{x}_P) \quad (2.1)$$

, where ϕ_P represents the value at the position \mathbf{x}_P . Generally, the gradient of ϕ is approximated by fitting some nearby sampled points, i.e., solving the following set of equations

$$\mathbf{d}_n \cdot (\nabla\phi)_P = \phi_n - \phi_P \quad (n = 1, \dots, N) \quad (2.2)$$

, in which $\mathbf{d}_n = \mathbf{x}_n - \mathbf{x}_P$ is the displacement vector from point \mathbf{x}_P to one of its neighbor points \mathbf{x}_n . The least-square method is used to solve this set of equations.

$$\mathbf{d}_n^T \mathbf{d}_n \cdot (\nabla\phi)_P = \mathbf{d}_n^T (\phi_n - \phi_P) \quad (2.3)$$

$$\sum_{n=1}^N \mathbf{d}_n^T \mathbf{d}_n (\nabla\phi)_P = \sum_{n=1}^N \mathbf{d}_n^T (\phi_n - \phi_P) \quad (2.4)$$

Utilizing a weighting function $w_n = 1/|\mathbf{d}_n|$ gives

$$\sum_{n=1}^N w_n^2 \mathbf{d}_n^T \mathbf{d}_n (\nabla\phi)_P = \sum_{n=1}^N w_n^2 \mathbf{d}_n^T (\phi_n - \phi_P) \quad (2.5)$$

For brevity, $\mathbf{M} = \sum_{n=1}^N w_n^2 \mathbf{d}_n^T \mathbf{d}_n$ and $\mathbf{f} = \sum_{n=1}^N w_n^2 \mathbf{d}_n^T (\phi_n - \phi_P)$ are defined. Finally, it is

$$\mathbf{M}(\nabla\phi)_P = \mathbf{f}. \quad (2.6)$$

Since \mathbf{M} and \mathbf{f} are known, the gradient $(\nabla\phi)_P$ can be calculated. Three kinds of least square gradient methods are implemented in OpenFOAM®. The major difference lies in the selection of the sampling cells, i.e., the stencil. When selecting the entry `leastSquaresGrad`, the face-neighbor cells are the stencil. The `pointCellsLeastSquares` uses the point-neighbor cells as the stencil, whereas applying `edgeCellsLeastSquares` regards the edge-neighbor cells as the stencil. Jasak and Weller [166] mentioned that this gradient calculation method can promise second-order accuracy regardless of the sampled points.

In interIsoFoam, Scheufler and Roenby [49] adopted a slightly different method to approximate the gradient. The starting point is a first degree polynomial function

$$\hat{\phi}_n = a_0 + a_1 d_{n,x} + a_2 d_{n,y} + a_3 d_{n,z} \quad (2.7)$$

, where $\hat{\phi}_n$ represents the approximated ϕ_n , $d_{n,x}$, $d_{n,y}$ and $d_{n,z}$ are three components of \mathbf{d}_n , a_0, a_1, a_2, a_3 are four polynomial parameters. $e_N = \sum_{n=1}^N (\phi_n - \hat{\phi}_n)^2$ is defined to describe the residual. The aim of this method is to calculate a_0, a_1, a_2, a_3 to minimize the e_N . The partial derivative of e_N are

$$\begin{aligned} \frac{\partial e_N}{\partial a_0} &= -2 \sum_{n=1}^N [\phi_n - (a_0 + a_1 d_{n,x} + a_2 d_{n,y} + a_3 d_{n,z})] = 0 \\ \frac{\partial e_N}{\partial a_1} &= -2 \sum_{n=1}^N [\phi_n - (a_0 + a_1 d_{n,x} + a_2 d_{n,y} + a_3 d_{n,z})] d_{n,x} = 0 \\ \frac{\partial e_N}{\partial a_2} &= -2 \sum_{n=1}^N [\phi_n - (a_0 + a_1 d_{n,x} + a_2 d_{n,y} + a_3 d_{n,z})] d_{n,y} = 0 \\ \frac{\partial e_N}{\partial a_3} &= -2 \sum_{n=1}^N [\phi_n - (a_0 + a_1 d_{n,x} + a_2 d_{n,y} + a_3 d_{n,z})] d_{n,z} = 0. \end{aligned} \quad (2.8)$$

A matrix can be assembled from the equation eq. (2.8)

$$\mathbf{M}' \mathbf{p} = \mathbf{f}' \quad (2.9)$$

, where \mathbf{M}' is a 4×4 matrix that can be represented by $\mathbf{M}' = \sum_{n=1}^N \mathbf{d}'^T \mathbf{d}'$, $\mathbf{d}' = (1, d_{n,x}, d_{n,y}, d_{n,z})$, and $\mathbf{f}' = \sum_{n=1}^N \mathbf{d}'^T \phi_n$. The \mathbf{p} is the polynomial parameters vector. From the equation eq. (2.1), it is clear that the last three components of \mathbf{g} compose the gradient of ϕ .

3 Correct cyclic boundary condition for the plicRDF-isoAdvector method

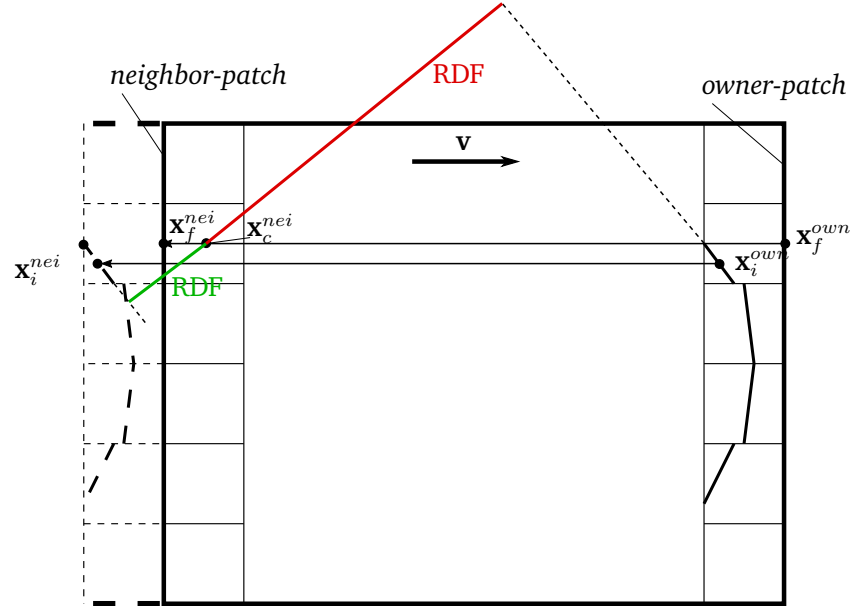


Figure 59: The ghost cells layer to correct signed distance: the green line (—) presents fixed RDF from cell centroid x_c^{nei} to interface in the cyclic neighbor cell, while the red line (—) depicts the original unfixed RDF.

A cyclic boundary condition (BC) treats two boundary patches as if they were physically connected, with their respective cell layers placed next to each other. For the geometrical VOF method such as the plicRDF-isoAdvector, the cyclic boundary condition impacts the interface reconstruction and the volume fraction advection. To achieve this, the cyclic BC performs calculations on the so-called *owner-patch*, and then reflects the result to the so-called *neighbor-patch*, as shown schematically in Figure 59 for a geometric VOF method in two dimensions.

For the advection discretization at cyclic BC, we noticed that the fluxed phase-specific volumes V_f^α field is not properly adjusted for the cyclic patches.

In the first step, the cyclic BC initializes V_f^α at every face of both cyclic patches using the upwind scheme, i.e.,

$$V_f^{\alpha,init} = F_f \alpha_U \Delta t, \quad (3.1)$$

where $F_f := \mathbf{v}_f \cdot \mathbf{S}_f$ is the volumetric flux at the centroid of the cell-face S_f , α_U is the volume fraction of the upwind cell U w.r.t the cell face S_f , and Δt is the time step.

In the second step, the cyclic BC computes the geometric V_f^α at the faces that belong to interface cells on both patches

$$V_f^\alpha = \int_{t^n}^{t^{n+1}} \int_{S_f} \chi \mathbf{v} \cdot \mathbf{n} dS dt = \int_{t^n}^{t^{n+1}} \frac{F_f(t)}{|\mathbf{S}_f|} A_f(t) dt \quad (3.2)$$

The existing cyclic BC [145] does not consider cyclic boundary conditions. The discretization in cell layers adjacent to two cyclic boundary patches should handle the cell layers as if they are placed next to each other as shown in Figure 59. A phase-specific volume fluxed out of the domain on a face that belongs to the cyclic owner-patch, should be fluxed into the corresponding face in the cyclic-neighbor patch, as described in Algorithm 10.

Algorithm 10 The modified fluxed phase-specific volumes V_f^α update method in interIsoFoam.

- 1: Initialize $V_f^{\alpha,owner}$ using the upwind scheme. ▷ Equation (3.1)
 - 2: **for** all boundary patches **do**
 - 3: **if** boundary patch is cyclic **then**
 - 4: **for** all cyclic-patch faces $f \in [1, |P_{cyclic}|]$ **do**
 - 5: **if** $F_f > 0$ **then**
 - 6: Geometrically compute the outflow V_f^α .
 - 7: **else if** $F_f < 0$ **then** ▷ The inflow $F_f < 0$ here is outflow $F_f > 0$ of the neighbor.
 - 8: $V_f^\alpha = -V_f^{\alpha,neighbor}$
 - 9: **end if**
 - 10: **end for**
 - 11: **end if**
 - 12: **end for**
-

Aside from the consistent calculation of V_f^α in the advection, the cyclic BC also affects the geometric interface reconstruction. The plicRDF reconstruction [6] uses the reconstructed

distance function (RDF) and its gradient for improving discrete interface-normal vectors. When considering the cyclic boundary condition, the distance calculation should be treated carefully, especially in the case where two cells sharing a vertex are also attached to two cyclic patches. This issue is illustrated in Figure 59, where \mathbf{x}_i^{own} denotes a center of a VOF interface polygon located in a corresponding interface cell Ω_i^{own} that belongs to the cell layer attached to the owner patch of the cyclic BC. In the plicRDF implementation in OpenFOAM-v2306 [145], the cyclic BC falsely uses VOF interface polygon centers \mathbf{x}_i^{own} to compute signed distances at centers \mathbf{x}_c^{nei} in the cell layer adjacent to the cyclic neighbor-patch. A correct implementation of the cyclic BC requires positions \mathbf{x}_i^{nei} , as shown in Figure 59. The VOF interface centroids from the cell layer adjacent to the cyclic owner-patch \mathbf{x}_i^{own} and the face centers of the cyclic owner-patch can be used to compute \mathbf{x}_i^{nei} for every \mathbf{x}_i^{own} and facilitate a correct cyclic (periodic) computation of the Reconstructed Distance Function (RDF) in the plicRDF-isoAdvector method.

We define a transformation of the position of the interface centers using

$$\mathbf{x}_i^{nei} := \mathbf{x}_i^{own} + (\mathbf{x}_f^{nei} - \mathbf{x}_f^{own}), \quad (3.3)$$

where $(\mathbf{x}_f^{nei} - \mathbf{x}_f^{own})$ is the difference (displacement, or transformation) vector between the face centers of cyclic BC owner and neighbor patch face centers.

Note that there is no need to transform the interface normals \mathbf{n}_i , because they are orientation and not position vectors. The transformed PLIC polygon centroids together with the cyclic BC neighbor-patch interface normals $\mathbf{n}_i^{nei} = \mathbf{n}_i^{own}$ build a cyclic ghost-data layer that is shown schematically by dashed cells in Figure 59.

Signed distances in the cell layer adjacent to the cyclic owner-patch at cell centers \mathbf{x}_i^{own} , are thus computed from the transformed PLIC interface information from the cyclic patch-owner data.

If the cyclic-patch-adjacent cell Ω_c does contain its own PLIC interface with the PLIC centroid, then the signed distance to this PLIC interface is used, if it is closer to \mathbf{x}_i of that cell.

We verify our discretization of the cyclic BC in the plicRDF-isoAdvector method using a constant flow case shown in Figure 60, where both the droplet and the ambient flow have the same initial velocity $\mathbf{v} = (v_x, 0)$. The left and right boundaries are set as cyclic (periodic). It is noteworthy that the momentum equation Equation (2.15) is not solved in this test case,

ensuring exactly constant velocity and pressure over time, aiming to only to test the periodic (cyclic) interface reconstruction and advection. We test two interface reconstruction methods, e.g. isoAlpha and plicRDF from [6].

Figure 61 shows the droplet reaching the right cyclic boundary. With the erroneous calculation of the fluxed phase-specific volume V_f^α at the cyclic boundaries, as shown in Figure 61a, the interfaces appear in the patch neighbor cells incorrectly. The Figure 61b show the accurate result of our modification with a single interface cell. The modification of V_f^α impacts both isoAlpha and plicRDF methods, while adapting the displacement is crucial only for the plicRDF reconstruction. As shown in Figure 62a, without modifying the displacement vector, some liquid remains in the cell layer adjacent to the neighbor-patch after the droplet crosses the cyclic boundary, and reaches a location far from both cyclic boundaries. With applying our modification from Figure 62b, the droplet retains its initial form. The fixed cyclic boundary condition is available in [123].

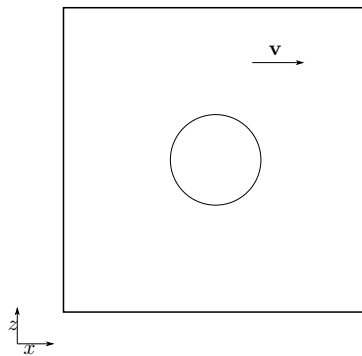


Figure 60: A droplet moves with the ambient constant flow.

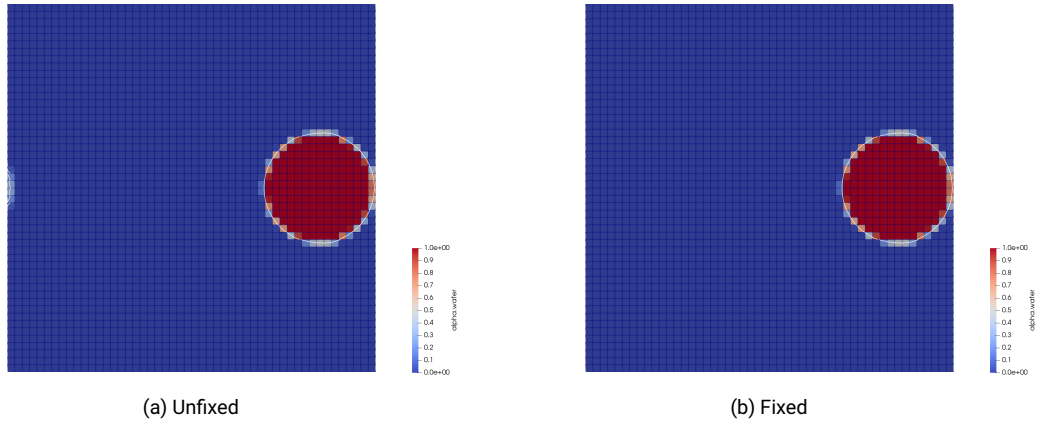


Figure 61: The alpha field and interfaces reconstructed by iso-Alpha method reach to the right cyclic boundary; blue region: the ambient flow; red region: the liquid droplet; white line segments: the PLIC interfaces.

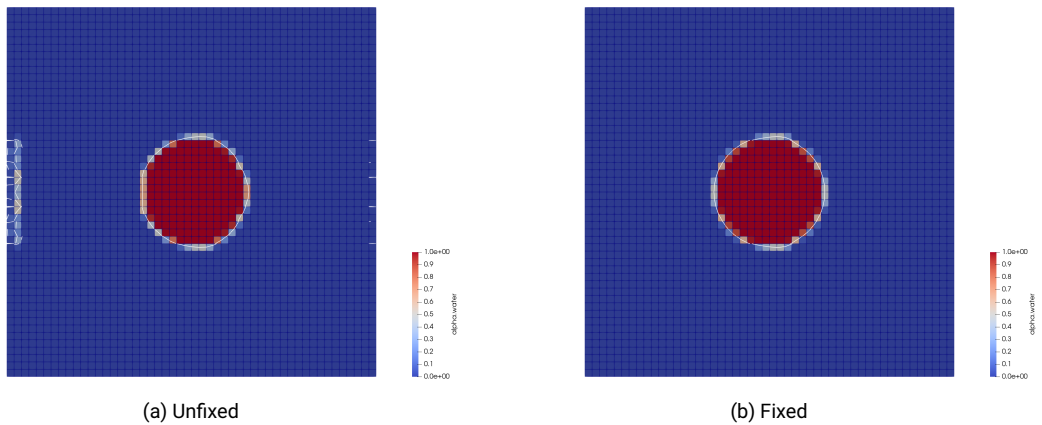


Figure 62: The alpha field and interfaces reconstructed by plic-RDF method cross the right cyclic boundary; blue region: the ambient flow; red region: the liquid droplet; white line segments: the PLIC interfaces.

4 Viscous term modelling

The viscous term in eq. (2.15) is split, as follows

$$-\nabla \cdot (\mu(\nabla \mathbf{v} + (\nabla \mathbf{v})^T)) = -\nabla \cdot (\mu \nabla \mathbf{v}) - \nabla \cdot (\mu(\nabla \mathbf{v})^T) \quad (4.1)$$

In this thesis, the first term on r.h.s of eq. (4.1) is treated implicitly, while the second term is calculated explicitly as a source term by directly interpolating $\mu(\nabla \mathbf{v})^T$ to cell faces with an arithmetic mean viscosity from eq. (3.62), i.e.

$$\nabla \cdot (\mu(\nabla \mathbf{v})^T) \approx \frac{1}{|\Omega_c|} \sum_{f \in F_c} \mu_f (\nabla \mathbf{v}^n)_f^T \cdot \mathbf{S}_f. \quad (4.2)$$

This treatment works well for the cases where the viscous effect does not dominate. However, in the viscous force dominated cases such as rising air bubble in aqueous sugar solutions, which is simulated in sections 3.5.4 and 6.4, this traditional treatment causes failures in section 3.5.4 and damped rising velocity at the acceleration stage in section 6.4 for the cases containing solution with higher viscosity.

One possible way to improve the approximation of eq. (4.1) involves further inspection on the explicit part in eq. (4.1), as follows

$$\begin{aligned} \nabla \cdot (\mu(\nabla \mathbf{v})^T) &= \nabla \mu \cdot (\nabla \mathbf{v})^T + \mu \nabla \cdot (\nabla \mathbf{v})^T \\ &= \nabla \mu \cdot (\nabla \mathbf{v})^T + \mu \nabla (\nabla \cdot \mathbf{v}). \end{aligned} \quad (4.3)$$

According to the continuity condition, i.e., $\nabla \cdot \mathbf{v} = 0$, the approximation of $\nabla \cdot (\mu(\nabla \mathbf{v})^T)$ can switch to

$$\nabla \cdot (\mu(\nabla \mathbf{v})^T) = \nabla \mu \cdot (\nabla \mathbf{v})^T \approx (\nabla \mu)_c \cdot (\nabla \mathbf{v})_c^T, \quad (4.4)$$

where the gradient at cell center can be approximated by utilizing Gauss-divergence theorem or Least-square method. Another way to deal with $\nabla \mu \cdot (\nabla \mathbf{v})^T$ explicitly is to put it in Poisson's equation as a source term. In this case, the divergence of $\nabla \mu \cdot (\nabla \mathbf{v})^T$ needs to be approximated, which has the formulation

$$\begin{aligned} \nabla \cdot (\nabla \mu \cdot (\nabla \mathbf{v})^T) &= \nabla \mathbf{v} : \nabla \nabla \mu + \nabla \mu \cdot \nabla (\nabla \cdot \mathbf{v}) \\ &= \nabla \mathbf{v} : \nabla \nabla \mu \\ &\approx (\nabla \mathbf{v})_c : (\nabla \nabla \mu)_c \end{aligned} \quad (4.5)$$

It is remarkable that $\nabla \cdot \mathbf{v} = 0$ can not be ensured strictly everywhere in computation, especially near the interface, where the continuity errors are usually visible. The errors are then amplified by the gradient calculation, which may have unpredictable effects on results. Thus, any omission of $\nabla(\nabla \cdot \mathbf{v})$ or $\nabla \cdot \mathbf{v}$ on the numerical level should be further discussed.

Another possible improvement may come from the careful selection of average method for viscosity. The simplest and most widespread method is arithmetic average, which is adopted in this thesis. Patankar [167] thought any diffusion coefficient including the viscosity should be handled in the same way as a conductivity, which has a harmonic formulation rather than the arithmetic formulation of average. The harmonic average of viscosity has been employed by many later works, e.g. [52, 53, 72, 168]. Deubelbeiss and Kaus [169] proposed a geometric average method and compared these three average methods. The results showed that the performances of these methods vary among different cases.

$$\begin{aligned}
 \mu_{arth} &= \mu^- \alpha + \mu^+ (1 - \alpha) \\
 \mu_{harm} &= \frac{\mu^- \mu^+}{\mu^- (1 - \alpha) + \mu^+ \alpha} \\
 \mu_{geom} &= (\mu^-)^\alpha (\mu^+)^{1-\alpha}
 \end{aligned} \tag{4.6}$$

To estimate viscosity, Coward et al. [170] put forward a blended method, which differentiates components in tensor $\nabla \mathbf{v}$ based on if a component is continuous or non-continuous across the interface. For the component that is continuous across interface, a arithmetic average of viscosity is utilized to construct the corresponding stress tensor component, whereas the harmonic average is used for non-continuous component.

5 Gravity term modelling

As described in section 2, the body force $\rho\mathbf{g}$ is replaced by the interfacial force formulation $(\mathbf{g} \cdot \mathbf{x})\nabla\rho$ to limit the gravitational effects to the interfacial region, and to simplify the wall pressure boundary condition [139, 171]. Like parasitic currents appearing in the static droplet equilibrium when numerically solving the governing eq. (5.1), numerical errors are also generated in the case of hydrostatic equilibrium, whose governing equation is displayed by eq. (5.2)

$$-\nabla p + \sigma\kappa\nabla\alpha = 0 \quad (5.1)$$

$$-\nabla p - (\mathbf{g} \cdot \mathbf{x})\nabla\rho = 0. \quad (5.2)$$

Two primary sources have been identified as the sources of spurious currents: inaccurate curvature estimation and discrepancies in discretization between the surface tension and pressure forces, as noted in previous studies [139, 172]. Many works like [139] (*and more, check the collected papers later*) noticed the generality of adopting identical discretization for pressure gradient and the interfacial force and extended it to maintain hydrostatic equilibrium between pressure gradient and gravity force. However, fewer works discussed the impact of approximating the term $(\mathbf{g} \cdot \mathbf{x})$ on the equilibrium. In fact, the errors arising from carelessly estimating $(\mathbf{g} \cdot \mathbf{x})$ can be large, which will be explained and verified in the following. To reduce the errors, Montazeri [162] put forward that the real altitude of the points, where the pressure Poisson equation is solved, in the interfacial region i.e., interfacial cell face centers for staggered grids the author used, should be replaced by the altitude of the closest point at the interface. In [162], the level-set method is used to capture the interface on the staggered grid. The distances' vectors from the face centers of interfacial cells to the interface are known and projected to the gravitational direction. The altitude of the closest point at the interface can be obtained by adding the projected altitude and the altitude of the targeted point. A series of subsequent works (Montazeri, Bussmann, and Mostaghimi [163] and Montazeri and Ward [164]) showed promising results in the hydrostatic equilibrium case when using the corrected altitude. Møller [165] adopted a similar approach to correct the altitude, whose work is based on a PLIC-VOF method isoAdvector developed on collocated unstructured mesh [36]. In Møller [165], the closest points of the centers of the interfacial cells and their

point-neighbor non-interfacial cells are located at the interface and then assigned to the corresponding center to update pressure. Although the correction is utilized, none of the mentioned works [162, 165] elaborated on it at full length.

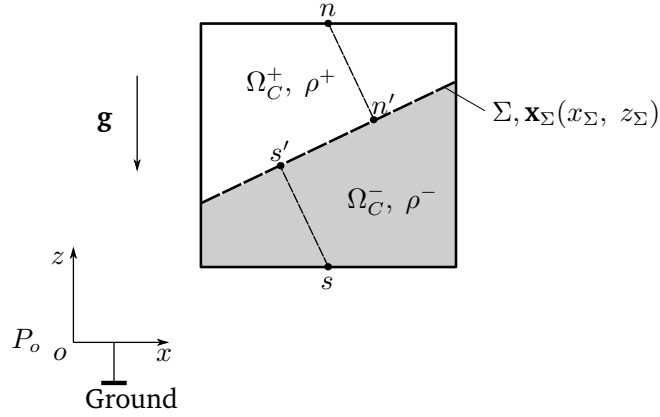


Figure 63: A cell Ω_C is separated by the interface Σ into two regions: the shaded region Ω_C^- is occupied by the flow with density ρ^- , the rest region Ω_C^+ is occupied by another flow with density ρ^+ .

As shown in fig. 63, it is assumed that the origin of the global coordinate system o is connected to the ground, where the pressure is set to P_o , and the gravitational acceleration \mathbf{g} is in the $-z$ direction. The total pressure of any point \mathbf{x} located in the cell Ω_C has the following formulation

$$P(x, z) = \begin{cases} P_o + \rho^- gz & , \quad z < z_\Sigma \\ P_o + \rho^- gz_\Sigma + \rho^+ g(z - z_\Sigma) & , \quad z > z_\Sigma. \end{cases} \quad (5.3)$$

According to the definition of dynamic pressure in section 2.2, the corresponding dynamic pressure at \mathbf{x} is defined by

$$p(x, z) = \begin{cases} P_o & , \quad z < z_\Sigma \\ P_o + (\rho^- - \rho^+)gz_\Sigma & , \quad z > z_\Sigma. \end{cases} \quad (5.4)$$

The analytical pressure jump $\Delta p = (\rho^- - \rho^+)gz_\Sigma$ varies along the interface in the 2D-case of




fig. 63. For the isoAdvector method used in this work, the interface plane is reconstructed every time step in a cell layer, i.e., the interfacial cell layer.

Bibliography

- [1] David L Youngs. “Time-dependent multi-material flow with large fluid distortion”. In: *Numerical methods for fluid dynamics* (1982).
- [2] Tobias Tolle, Dieter Bothe, and Tomislav Marić. “SAAMPLE: A Segregated Accuracy-driven Algorithm for Multiphase Pressure-Linked Equations”. In: *Computers & Fluids* 200 (2020), p. 104450. URL: <https://doi.org/10.1016/j.compfluid.2020.104450>.
- [3] D. Bhaga and M. E. Weber. “Bubbles in viscous liquids: shapes, wakes and velocities”. In: *Journal of Fluid Mechanics* 105 (1981), pp. 61–85. DOI: 10.1017/S002211208100311X.
- [4] Jinsong Hua and Jing Lou. “Numerical simulation of bubble rising in viscous liquid”. en. In: *Journal of Computational Physics* 222.2 (Mar. 2007), pp. 769–795. ISSN: 00219991. DOI: 10.1016/j.jcp.2006.08.008. URL: <https://linkinghub.elsevier.com/retrieve/pii/S0021999106003949> (visited on 01/25/2023).
- [5] G.R. Anjos, N. Borhani, N. Mangiavacchi, and J.R. Thome. “A 3D moving mesh Finite Element Method for two-phase flows”. In: *Journal of Computational Physics* 270 (2014), pp. 366–377. ISSN: 0021-9991. URL: <https://www.sciencedirect.com/science/article/pii/S0021999114002551>.
- [6] Henning Scheufler and Johan Roenby. “Accurate and efficient surface reconstruction from volume fraction data on general meshes”. In: *Journal of computational physics* 383 (2019), pp. 1–23.

-
-
- [7] Davide Zuzio, Annagrazia Orazzo, Jean Luc Estivalèzes, and Isabelle Lagrange. “A new efficient momentum preserving Level-Set/VOF method for high density and momentum ratio incompressible two-phase flows”. In: *Journal of Computational Physics* 410 (2020), p. 109342. ISSN: 10902716. DOI: 10.1016/j.jcp.2020.109342.
- [8] Xiaoyi Li and Marios C Soteriou. “High fidelity simulation and analysis of liquid jet atomization in a gaseous crossflow at intermediate Weber numbers”. In: *Physics of Fluids* 28.8 (Aug. 2016), p. 082101. ISSN: 1070-6631. DOI: 10.1063/1.4959290. URL: <http://aip.scitation.org/doi/10.1063/1.4959290>.
- [9] Bernhard Godderidge, Stephen Turnock, Mingyi Tan, and Chris Earl. “An investigation of multiphase CFD modelling of a lateral sloshing tank”. In: *Computers & Fluids* 38.2 (2009), pp. 183–193. URL: <https://doi.org/10.1016/j.compfluid.2007.11.007>.
- [10] S Soukane and F Trochu. “Application of the level set method to the simulation of resin transfer molding”. In: *Composites Science and Technology* 66.7-8 (2006), pp. 1067–1080. URL: <https://doi.org/10.1016/j.compscitech.2005.03.001>.
- [11] Zhiliang Gao, Qiuxin Gao, and Dracos Vassalos. “Numerical simulation of flooding of a damaged ship”. In: *Ocean Engineering* 38.14-15 (2011), pp. 1649–1662. URL: <https://doi.org/10.1016/j.oceaneng.2011.07.020>.
- [12] Jianhua Chen, Charitos Anastasiou, Sibö Cheng, Nausheen Mehboob Basha, Lyes Kahouadji, Rossella Arcucci, Panagiota Angeli, and Omar K Matar. “Computational fluid dynamics simulations of phase separation in dispersed oil-water pipe flows”. In: *Chemical Engineering Science* 267 (2023), p. 118310. URL: <https://doi.org/10.1016/j.ces.2022.118310>.
- [13] Vernon A Squire, John P Dugan, Peter Wadhams, Philip J Rottier, and Antony K Liu. “Of ocean waves and sea ice”. In: *Annual Review of Fluid Mechanics* 27.1 (1995), pp. 115–168. URL: <https://doi.org/10.1146/annurev.fl.27.010195.000555>.

-
-
- [14] Luise Nagel, Anja Lippert, Tobias Tolle, Ronny Leonhardt, Huijie Zhang, and Tomislav Marić. “Stabilizing the unstructured Volume-of-Fluid method for capillary flows in microstructures using artificial viscosity”. In: *Experimental and Computational Multiphase Flow* 6.2 (2024), pp. 140–153. URL: <https://doi.org/10.1007/s42757-023-0181-y>.
- [15] Huijie Zhang, Anja Lippert, Ronny Leonhardt, Tobias Tolle, Luise Nagel, Mathis Fricke, and Tomislav Marić. “Experimental study of dynamic wetting behavior through curved microchannels with automated image analysis”. In: *Experiments in Fluids* 65.6 (2024), p. 95. URL: <https://doi.org/10.1007/s00348-024-03828-7>.
- [16] Joel H Ferziger, Milovan Perić, and Robert L Street. *Computational methods for fluid dynamics*. Vol. 4. Springer, 2020, pp. 93–97. URL: <https://link.springer.com/book/10.1007/978-3-319-99693-6>.
- [17] Fadl Moukalled, L Mangani, Marwan Darwish, et al. *The finite volume method in computational fluid dynamics: An Advanced Introduction with OpenFOAM® and Matlab®*. Vol. 6. Springer, 2016.
- [18] Andrea Prosperetti and Grétar Tryggvason. *Computational methods for multiphase flow*. Cambridge university press, 2009. URL: <https://doi.org/10.1017/CB09780511607486>.
- [19] Tomislav Marić, Jens Höpken, and Kyle G. Mooney. *The OpenFOAM Technology Primer*. Zenodo, Mar. 2021. URL: <https://doi.org/10.5281/zenodo.4630596>.
- [20] Charles Hirsch. *Numerical computation of internal and external flows: The fundamentals of computational fluid dynamics*. Elsevier, 2007. URL: <https://doi.org/10.1016/B978-0-7506-6594-0.X5037-1>.
- [21] G. Tryggvason, A. Fernández, A. Esmaeeli, and B. Bunner. *Direct numerical simulations of gas – liquid multiphase flows*. Vol. 66. 2011, pp. 313–338. ISBN: 1402005237. DOI: 10.1017/CB09780511975264. URL: <https://doi.org/10.1017/CB09780511975264>.
- [22] Christopher Greenshields and Henry Weller. *Notes on Computational Fluid Dynamics: General Principles*. Reading, UK: CFD Direct Ltd, 2022.

-
-
- [23] Jiyuan Tu, Guan Heng Yeoh, Chaoqun Liu, and Yao Tao. *Computational fluid dynamics: a practical approach*. Elsevier, 2019. URL: <https://doi.org/10.1016/C2015-0-06135-4>.
- [24] Nicholas P Waterson and Herman Deconinck. “Design principles for bounded higher-order convection schemes—a unified approach”. In: *Journal of Computational Physics* 224.1 (2007), pp. 182–207. URL: <https://doi.org/10.1016/j.jcp.2007.01.021>.
- [25] Vladimir D Liseikin. *Grid generation methods*. Vol. 1. Springer, 1999.
- [26] Henk Kaarle Versteeg and Weeratunge Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.
- [27] Jeremiah U Brackbill, Douglas B Kothe, and Charles Zemach. “A continuum method for modeling surface tension”. In: *Journal of computational physics* 100.2 (1992), pp. 335–354. URL: [https://doi.org/10.1016/0021-9991\(92\)90240-Y](https://doi.org/10.1016/0021-9991(92)90240-Y).
- [28] Hrvoje Jasak. “Error analysis and estimation for the finite volume method with applications to fluid flows.” PhD thesis. Imperial College London, 1996.
- [29] Juretic Franjo. “Error Analysis in Finite Volume CFD”. PhD thesis. Imperial College London, 2004.
- [30] Suhas V Patankar and D Brian Spalding. “A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows”. In: *International journal of heat and mass transfer* 15.10 (1972), pp. 1787–1806.
- [31] Raad I Issa. “Solution of the implicitly discretised fluid flow equations by operator-splitting”. In: *Journal of computational physics* 62.1 (1986), pp. 40–65.
- [32] Tobias Holzmann. *Mathematics, Numerics, Derivations and OpenFOAM®*. Holzmann CFD, Leoben, fourth edition, 2016.
- [33] Shahab Mirjalili, Suhas S Jain, and Micheal Dodd. “Interface-capturing methods for two-phase flows: An overview and recent developments”. In: *Center for Turbulence Research Annual Research Briefs* 2017.117-135 (2017), p. 13. URL: <https://doddm.com/publications/2017-ctr-sm-sj-md.pdf>.

-
-
- [34] Tomislav Marić, Douglas B Kothe, and Dieter Bothe. “Unstructured un-split geometrical volume-of-fluid methods—a review”. In: *Journal of Computational Physics* 420 (2020), p. 109695. URL: <https://doi.org/10.1016/j.jcp.2020.109695>.
- [35] Tomislav Marić, Holger Marschall, and Dieter Bothe. “lentFoam—A hybrid Level Set/Front Tracking method on unstructured meshes”. In: *Computers & Fluids* 113 (2015), pp. 20–31. URL: <https://doi.org/10.1016/j.compfluid.2014.12.019>.
- [36] Johan Roenby, Henrik Bredmose, and Hrvoje Jasak. “A computational method for sharp interface advection”. In: *Royal Society Open Science* 3.11 (2016). ISSN: 20545703. DOI: 10.1098/rsos.160405. eprint: 1601.05392. URL: <https://arxiv.org/abs/1601.05392>.
- [37] Grétar Tryggvason, Bernard Bunner, Asghar Esmaeeli, Damir Juric, N Al-Rawahi, W Tauber, J Han, S Nas, and Y-J Jan. “A front-tracking method for the computations of multiphase flow”. In: *Journal of computational physics* 169.2 (2001), pp. 708–759. URL: <https://doi.org/10.1006/jcph.2001.6726>.
- [38] Seungwon Shin and Damir Juric. “Modeling three-dimensional multiphase flow using a level contour reconstruction method for front tracking without connectivity”. In: *Journal of Computational Physics* 180.2 (2002), pp. 427–470. URL: <https://doi.org/10.1006/jcph.2002.7086>.
- [39] Hiroshi Terashima and Grétar Tryggvason. “A front-tracking/ghost-fluid method for fluid interfaces in compressible flows”. In: *Journal of Computational Physics* 228.11 (2009), pp. 4012–4037. URL: <https://doi.org/10.1016/j.jcp.2009.02.023>.
- [40] Pierre Trontin, Stéphane Vincent, Jean-Luc Estivalezes, and Jean-Paul Caltagirone. “A subgrid computation of the curvature by a particle/level-set method. Application to a front-tracking/ghost-fluid method for incompressible flows”. In: *Journal of Computational Physics* 231.20 (2012), pp. 6990–7010. URL: <https://doi.org/10.1016/j.jcp.2012.07.002>.

-
-
- [41] Xu-Dong Liu, Ronald P Fedkiw, and Myungjoo Kang. “A boundary condition capturing method for Poisson’s equation on irregular domains”. In: *Journal of computational Physics* 160.1 (2000), pp. 151–178. URL: <https://doi.org/10.1006/jcph.2000.6444>.
- [42] Myungjoo Kang, Ronald P Fedkiw, and Xu-Dong Liu. “A boundary condition capturing method for multiphase incompressible flow”. In: *Journal of Scientific Computing* 15 (2000), pp. 323–360. URL: <https://doi.org/10.1023/A:1011178417620>.
- [43] Stanley Osher and James A Sethian. “Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations”. In: *Journal of computational physics* 79.1 (1988), pp. 12–49. URL: [https://doi.org/10.1016/0021-9991\(88\)90002-2](https://doi.org/10.1016/0021-9991(88)90002-2).
- [44] Mark Sussman, Peter Smereka, and Stanley Osher. “A level set approach for computing solutions to incompressible two-phase flow”. In: *Journal of Computational physics* 114.1 (1994), pp. 146–159. URL: <https://doi.org/10.1006/jcph.1994.1155>.
- [45] Stanley Osher, Ronald Fedkiw, and K Piechor. “Level set methods and dynamic implicit surfaces”. In: *Appl. Mech. Rev.* 57.3 (2004), B15–B15. URL: <http://ndl.ethernet.edu.et/bitstream/123456789/33018/1/9.pdf.pdf>.
- [46] Seungwon Shin, SI Abdel-Khalik, Virginie Daru, and Damir Juric. “Accurate representation of surface tension using the level contour reconstruction method”. In: *Journal of Computational Physics* 203.2 (2005), pp. 493–516. URL: <https://doi.org/10.1016/j.jcp.2004.09.003>.
- [47] Jun Liu, Tobias Tolle, Dieter Bothe, and Tomislav Marić. “An unstructured finite-volume Level Set / Front Tracking method for two-phase flows with large density-ratios”. In: *Journal of Computational Physics* (2023), p. 112426. ISSN: 0021-9991. URL: <https://doi.org/10.1016/j.jcp.2023.112426>.
- [48] Hector D Cenicerros, Alexandre M Roma, Aristeu Silveira-Neto, Millena M Villar, et al. “A robust, fully adaptive hybrid level-set/front-tracking method for two-phase flows with an accurate surface tension computation”. In: *Commun Comput Phys* 8.1 (2010),

-
- pp. 51–94. URL: http://global-sci.org/intro/article_detail/cicp/7563.html.
- [49] Henning Scheufler and Johan Roenby. “TwoPhaseFlow: A Framework for Developing Two Phase Flow Solvers in OpenFOAM”. In: *OpenFOAM® Journal* 3 (2023), pp. 200–224. URL: <https://doi.org/10.51560/ofj.v3.80>.
- [50] O Desjardins and V Moureau. “Methods for multiphase flows with high density ratio”. In: *Center for Turbulence Research Proceedings of the Summer Program* (2010), pp. 313–322. URL: https://web.stanford.edu/group/ctr/Summer/SP10/6_02_desjardins.pdf.
- [51] S. Ghods and M. Herrmann. “A consistent rescaled momentum transport method for simulating large density ratio incompressible multiphase flows using level set methods”. In: *Physica Scripta* 88.T155 (2013). ISSN: 00318949. DOI: 10.1088/0031-8949/2013/T155/014050. URL: <https://iopscience.iop.org/article/10.1088/0031-8949/2013/T155/014050>.
- [52] Nishant Nangia, Boyce E. Griffith, Neelesh A. Patankar, and Amneet Pal Singh Bhalla. “A robust incompressible Navier-Stokes solver for high density ratio multiphase flows”. In: *Journal of Computational Physics* 390 (2019), pp. 548–594. ISSN: 10902716. DOI: 10.1016/j.jcp.2019.03.042. eprint: 1809.01008.
- [53] Mehdi Raessi and Heinz Pitsch. “Consistent mass and momentum transport for simulating incompressible interfacial flows with large density ratios using the level set method”. In: *Computers and Fluids* 63 (2012), pp. 70–81. ISSN: 00457930. DOI: 10.1016/j.compfluid.2012.04.002.
- [54] Annagrazia Orazzo, Jean Luc Estivalezes, Isabelle Lagrange, and Davide Zuzio. “A vof-based consistent mass-momentum transport for two-phase flow simulations”. In: vol. 1C-2017. American Society of Mechanical Engineers, July 2017, pp. 1–11. ISBN: 9780791858066. DOI: 10.1115/FEDSM2017-69190.
- [55] Vincent Le Chenadec and Heinz Pitsch. “A 3D unsplit Forward/Backward Volume-of-Fluid approach and coupling to the level set method”. In: *J. Comput. Phys.* 233.1 (2013), pp. 10–33. ISSN: 00219991. DOI: 10.1016/j.jcp.2012.07.019.

-
-
- [56] *Unstructured Finite-Volume Level Set / Front Tracking - LENT code repository*. 2022. URL: <https://gitlab.com/leia-methods/lent/-/tree/2022-02-rhoLENT-R1>.
- [57] Murray Rudman. “A volume-tracking method for incompressible multifluid flows with large density variations”. In: *International Journal for Numerical Methods in Fluids* 28.2 (1998), pp. 357–378. ISSN: 02712091. DOI: 10.1002/(SICI)1097-0363(19980815)28:2<357::AID-FLD750>3.0.CO;2-D.
- [58] Markus Bussmann, Douglas B. Kothe, and James M. Sicilian. “Modeling high density ratio incompressible interfacial flows”. In: *American Society of Mechanical Engineers, Fluids Engineering Division (Publication) FED 257.1 B* (2002), pp. 707–713. DOI: 10.1115/FEDSM2002-31125. URL: <https://doi.org/10.1115/FEDSM2002-31125>.
- [59] William J Rider and Douglas B Kothe. “Reconstructing volume tracking”. In: *Journal of computational physics* 141.2 (1998), pp. 112–152. URL: <https://doi.org/10.1006/jcph.1998.5906>.
- [60] Gregor Cerne, Stojan Petelin, and Iztok Tiselj. “Numerical errors of the volume-of-fluid interface tracking algorithm”. In: *International journal for numerical methods in fluids* 38.4 (2002), pp. 329–350.
- [61] Mark Sussman, Kayne M Smith, M Yousuff Hussaini, Mitsuhiro Ohta, and R Zhi-Wei. “A sharp interface method for incompressible two-phase flows”. In: *Journal of computational physics* 221.2 (2007), pp. 469–505.
- [62] Mark Sussman and Elbrige Gerry Puckett. “A Coupled Level Set and Volume-of-Fluid Method for Computing 3D and Axisymmetric Incompressible Two-Phase Flows”. In: *Journal of Computational Physics* 162.2 (2000), pp. 301–337. ISSN: 00219991. DOI: 10.1006/jcph.2000.6537.
- [63] Vincent Le Chenadec and Heinz Pitsch. “A monotonicity preserving conservative sharp interface flow solver for high density ratio two-phase flows”. In: *Journal of Computational Physics* 249 (2013), pp. 185–203. ISSN: 10902716. DOI: 10.1016/j.jcp.2013.04.027.

-
-
- [64] Andrea Prosperetti. “Motion of two superposed viscous fluids”. In: *The Physics of Fluids* 24.7 (1981), pp. 1217–1223.
- [65] Geoffroy Vaudor, Alain Berlemont, Thibaut Ménard, and Mathieu Doring. “A consistent mass and momentum flux computation method using rudman-type technique with a clsvof solver”. In: *Fluids Engineering Division Summer Meeting*. Vol. 46230. American Society of Mechanical Engineers. 2014, V01CT23A012. URL: <https://doi.org/10.1115/FEDSM2014-21802>.
- [66] Thibault Ménard, Sebastien Tanguy, and Alain Berlemont. “Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet”. In: *International Journal of Multiphase Flow* 33.5 (2007), pp. 510–524. URL: <https://doi.org/10.1016/j.ijmultiphaseflow.2006.11.001>.
- [67] Wojciech Aniszewski, Thibaut Ménard, and Maciej Marek. “Volume of Fluid (VOF) type advection methods in two-phase flow: A comparative study”. In: *Computers & Fluids* 97 (2014), pp. 52–73. URL: <https://doi.org/10.1016/j.compfluid.2014.03.027>.
- [68] G. Vaudor, T. Ménard, W. Aniszewski, M. Doring, and A. Berlemont. “A consistent mass and momentum flux computation method for two phase flows. Application to atomization process”. In: *Computers and Fluids* 152 (2017), pp. 204–216. ISSN: 00457930. DOI: 10.1016/j.compfluid.2017.04.023. URL: <https://doi.org/10.1016/j.compfluid.2017.04.023>.
- [69] Mark Owkes and Olivier Desjardins. “A mass and momentum conserving unsplit semi-Lagrangian framework for simulating multiphase flows”. In: *Journal of Computational Physics* 332 (2017), pp. 21–46. ISSN: 10902716. URL: <http://dx.doi.org/10.1016/j.jcp.2016.11.046>.
- [70] Mark Owkes and Olivier Desjardins. “A computational framework for conservative, three-dimensional, unsplit, geometric transport with application to the volume-of-fluid (VOF) method”. In: *J. Comput. Phys.* 270 (2014), pp. 587–612. ISSN: 10902716. DOI: 10.1016/j.jcp.2014.04.022. URL: <http://dx.doi.org/10.1016/j.jcp.2014.04.022>.

-
- [71] Zixuan Yang, Min Lu, and Shizhao Wang. “A robust solver for incompressible high-Reynolds-number two-fluid flows with high density contrast”. In: *Journal of Computational Physics* (2021), p. 110474. URL: <https://doi.org/10.1016/j.jcp.2021.110474>.
- [72] Jitendra Kumar Patel and Ganesh Natarajan. “A novel consistent and well-balanced algorithm for simulations of multiphase flows on unstructured grids”. In: *Journal of Computational Physics* 350 (2017), pp. 207–236. ISSN: 10902716. URL: <http://dx.doi.org/10.1016/j.jcp.2017.08.047>.
- [73] Jai Manik, Amaresh Dalal, and Ganesh Natarajan. “A generic algorithm for three-dimensional multiphase flows on unstructured meshes”. In: *International Journal of Multiphase Flow* 106 (2018), pp. 228–242. ISSN: 03019322. URL: <https://doi.org/10.1016/j.ijmultiphaseflow.2018.04.010>.
- [74] MA Alves, PJ Oliveira, and FT Pinho. “A convergent and universally bounded interpolation scheme for the treatment of advection”. In: *International journal for numerical methods in fluids* 41.1 (2003), pp. 47–75. URL: <https://doi.org/10.1002/flid.428>.
- [75] Alexandre Joel Chorin. “The numerical solution of the Navier-Stokes equations for an incompressible fluid”. In: *Bulletin of the American Mathematical Society* 73.6 (1967), pp. 928–931. URL: <https://www.ams.org/bull/1967-73-06/S0002-9904-1967-11853-6/>.
- [76] T Arrufat, M Crialesi-Esposito, D Fuster, Y Ling, L Malan, S Pal, R Scardovelli, G Tryggvason, and S Zaleski. “A mass-momentum consistent, volume-of-fluid method for incompressible flow on staggered grids”. In: *Computers & Fluids* 215 (2021), p. 104785. URL: <https://doi.org/10.1016/j.compfluid.2020.104785>.
- [77] Hang Ding, Peter DM Spelt, and Chang Shu. “Diffuse interface model for incompressible two-phase flows with large density ratios”. In: *Journal of Computational Physics* 226.2 (2007), pp. 2078–2095. URL: <https://doi.org/10.1016/j.jcp.2007.06.028>.

-
-
- [78] Y. Wang, C. Shu, J. Y. Shao, J. Wu, and X. D. Niu. “A mass-conserved diffuse interface method and its application for incompressible multiphase flows with large density ratio”. In: *Journal of Computational Physics* 290 (2015), pp. 336–351. ISSN: 10902716. DOI: 10.1016/j.jcp.2015.03.005. URL: <https://doi.org/10.1016/j.jcp.2015.03.005>.
- [79] Ziyang Huang, Guang Lin, and Arezoo M. Ardekani. “Consistent, essentially conservative and balanced-force Phase-Field method to model incompressible two-phase flows”. In: *Journal of Computational Physics* 406 (2020), p. 109192. ISSN: 10902716. DOI: 10.1016/j.jcp.2019.109192. URL: <https://doi.org/10.1016/j.jcp.2019.109192>.
- [80] T Inamuro, Tajima Ogata, S Tajima, and N Konishi. “A lattice Boltzmann method for incompressible two-phase flows with large density differences”. In: *Journal of Computational physics* 198.2 (2004), pp. 628–644. URL: <https://doi.org/10.1016/j.jcp.2004.01.019>.
- [81] Taehun Lee and Ching-Long Lin. “A stable discretization of the lattice Boltzmann equation for simulation of incompressible two-phase flows at high density ratio”. In: *Journal of Computational Physics* 206.1 (2005), pp. 16–47. URL: <https://doi.org/10.1016/j.jcp.2004.12.001>.
- [82] HW Zheng, Chang Shu, and Yong-Tian Chew. “A lattice Boltzmann model for multiphase flows with large density ratio”. In: *Journal of computational physics* 218.1 (2006), pp. 353–371. URL: <https://doi.org/10.1016/j.jcp.2006.02.015>.
- [83] Seungwon Shin, Ikroh Yoon, and Damir Juric. “The local front reconstruction method for direct simulation of two-and three-dimensional multiphase flows”. In: *Journal of Computational Physics* 230.17 (2011), pp. 6605–6646. URL: <https://doi.org/10.1016/j.jcp.2011.04.040>.
- [84] Seungwon Shin, Jalel Chergui, and Damir Juric. “A solver for massively parallel direct numerical simulation of three-dimensional multiphase flows”. In: *Journal*

-
- of *Mechanical Science and Technology* 31.4 (2017), pp. 1739–1751. URL: <https://arxiv.org/abs/1410.8568>.
- [85] Tobias Tolle, Dirk Gründing, Dieter Bothe, and Tomislav Marić. “triSurfaceImmersion: Computing volume fractions and signed distances from triangulated surfaces immersed in unstructured meshes”. In: *Computer Physics Communications* 273 (2022), p. 108249. URL: <https://doi.org/10.1016/j.cpc.2021.108249>.
- [86] Christopher B Ivey and Parviz Moin. “Conservative and bounded volume-of-fluid advection on unstructured grids”. In: *J. Comput. Phys.* 350 (2017), pp. 387–419. ISSN: 10902716. URL: <http://dx.doi.org/10.1016/j.jcp.2017.08.054>.
- [87] Tomislav Marić, Holger Marschall, and Dieter Bothe. “An enhanced un-split face-vertex flux-based VoF method”. In: *Journal of computational physics* 371 (2018), pp. 967–993. URL: <https://doi.org/10.1016/j.jcp.2018.03.048>.
- [88] Metin Muradoglu and Gretar Tryggvason. “A front-tracking method for computation of interfacial flows with soluble surfactants”. In: *Journal of computational physics* 227.4 (2008), pp. 2238–2262. URL: <https://doi.org/10.1016/j.jcp.2007.10.003>.
- [89] Graham M Treece, Richard W Prager, and Andrew H Gee. “Regularised marching tetrahedra: improved iso-surface extraction”. In: *Computers & Graphics* 23.4 (1999), pp. 583–598. URL: [https://doi.org/10.1016/S0097-8493\(99\)00076-X](https://doi.org/10.1016/S0097-8493(99)00076-X).
- [90] Miles Detrixhe and Tariq D Aslam. “From level set to volume of fluid and back again at second-order accuracy”. In: *International Journal for Numerical Methods in Fluids* 80.4 (2016), pp. 231–255. URL: <https://doi.org/10.1002/flid.4076>.
- [91] Rajkeshar Singh and Wei Shyy. “Three-dimensional adaptive Cartesian grid method with conservative interface restructuring and reconstruction”. In: *Journal of Computational Physics* 224.1 (May 2007), pp. 150–167. ISSN: 00219991. URL: <https://linkinghub.elsevier.com/retrieve/pii/S002199910600622X> (visited on 02/04/2023).

-
- [92] Jinsong Hua, Jan F. Stene, and Ping Lin. “Numerical simulation of 3D bubbles rising in viscous liquids using a front tracking method”. en. In: *Journal of Computational Physics* 227.6 (Mar. 2008), pp. 3358–3382. ISSN: 00219991. URL: <https://linkinghub.elsevier.com/retrieve/pii/S0021999107005530> (visited on 01/26/2023).
- [93] M.R. Pivello, M.M. Villar, R. Serfaty, A.M. Roma, and A. Silveira-Neto. “A fully adaptive front tracking method for the simulation of two phase flows”. en. In: *International Journal of Multiphase Flow* 58 (Jan. 2014), pp. 72–82. ISSN: 03019322. URL: <https://linkinghub.elsevier.com/retrieve/pii/S0301932213001286> (visited on 01/26/2023).
- [94] Jun Liu, Tobias Tolle, and Tomislav Maric. *A consistent discretization of the single-field two-phase momentum convection term for the unstructured finite volume Level Set / Front Tracking method - data*. Version 2022-02-rhoLENT-R0. Zenodo, Feb. 2022. DOI: 10.5281/zenodo.6092417. URL: <https://doi.org/10.5281/zenodo.6092417>.
- [95] Jun Liu, Tobias Tolle, and Tomislav Maric. *A consistent discretization of the single-field two-phase momentum convection term for the unstructured finite volume Level Set / Front Tracking method - source code*. Version 2022-02-rhoLENT-R0. Feb. 2022. DOI: 10.5281/zenodo.6091930. URL: <https://doi.org/10.5281/zenodo.6091930>.
- [96] Jun Liu, Tobias Tolle, and Tomislav Maric. *LEvel Set / froNT tracking - LENT GitLab repository*. <https://gitlab.com/leia-methods/lent/-/tree/2022-02-rhoLENT-R1>. 2022.
- [97] Fabian Denner and Berend GM van Wachem. “Numerical time-step restrictions as a result of capillary waves”. In: *Journal of Computational Physics* 285 (2015), pp. 24–40. URL: <https://doi.org/10.1016/j.jcp.2015.01.021>.
- [98] Stéphane Popinet. “An accurate adaptive solver for surface-tension-driven interfacial flows”. In: *Journal of Computational Physics* 228.16 (2009), pp. 5838–5866. URL: <https://doi.org/10.1016/j.jcp.2009.04.042>.

-
-
- [99] Arthur W Adamson, Alice Petry Gast, et al. *Physical chemistry of surfaces, 6th Edition*. Vol. 150. New York: Wiley, 1997.
- [100] R Zivojnovic. *Silicone oil in vitreoretinal surgery*. Vol. 12. Springer Science & Business Media, 2012.
- [101] E Trinh and TG Wang. “Large-amplitude free and driven drop-shape oscillations: experimental observations”. In: *Journal of Fluid Mechanics* 122 (1982), pp. 315–338. DOI: 10.1017/S0022112082002237.
- [102] Horace Lamb. “Hydrodynamics”. In: (New York, Dover Publications, 1932).
- [103] Subrahmanyam Chandrasekhar. “The oscillations of a viscous liquid globe”. In: *Proceedings of the London Mathematical Society* 3.1 (1959), pp. 141–149.
- [104] CA Miller and LE Scriven. “The oscillations of a fluid droplet immersed in another fluid”. In: *Journal of fluid mechanics* 32.3 (1968), pp. 417–435.
- [105] A Prosperetti. “Normal-mode analysis for the oscillations of a viscous liquid drop in an immiscible liquid”. In: (1980).
- [106] WJ Hiller and TA Kowalewski. “Experimental analysis of free oscillating liquid drops”. In: *Proceedings of the 10th Australasian Fluid Mechanics Conference, University of Melbourne*. 1989, pp. 7–21.
- [107] Ziyang Huang, Guang Lin, and Arezoo M. Ardekani. “A mixed upwind/central WENO scheme for incompressible two-phase flows”. In: *Journal of Computational Physics* 387 (2019), pp. 455–480. ISSN: 10902716. DOI: 10.1016/j.jcp.2019.02.043. URL: <https://doi.org/10.1016/j.jcp.2019.02.043>.
- [108] Guang-Shan Jiang and Chi-Wang Shu. “Efficient implementation of weighted ENO schemes”. In: *Journal of computational physics* 126.1 (1996), pp. 202–228. DOI: 10.1006/jcph.1996.0130. URL: <https://doi.org/10.1006/jcph.1996.0130>.

-
-
- [109] Michael S Dodd and Antonino Ferrante. “A fast pressure-correction method for incompressible two-fluid flows”. In: *Journal of Computational Physics* 273 (2014), pp. 416–434. DOI: 10.1016/j.jcp.2014.05.024. URL: <https://doi.org/10.1016/j.jcp.2014.05.024>.
- [110] Bin Xie, Peng Jin, Yanping Du, and ShiJun Liao. “A consistent and balanced-force model for incompressible multiphase flows on polyhedral unstructured grids”. In: *International Journal of Multiphase Flow* 122 (2020), p. 103125. ISSN: 0301-9322. URL: <https://www.sciencedirect.com/science/article/pii/S0301932219304501>.
- [111] Bin Xie and Feng Xiao. “Toward efficient and accurate interface capturing on arbitrary hybrid unstructured grids: The THINC method with quadratic surface representation and Gaussian quadrature”. In: *Journal of Computational Physics* 349 (2017), pp. 415–440. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2017.08.028>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999117305995>.
- [112] Marianne M. Francois, Sharen J. Cummins, Edward D. Dendy, Douglas B. Kothe, James M. Sicilian, and Matthew W. Williams. “A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework”. In: *Journal of Computational Physics* 213.1 (2006), pp. 141–173. ISSN: 0021-9991. URL: <https://doi.org/10.1016/j.jcp.2005.08.004>.
- [113] Florian Desmons and Mathieu Coquerelle. “A generalized high-order momentum preserving (HOMP) method in the one-fluid model for incompressible two phase flows with high density ratio”. In: *Journal of Computational Physics* 437 (2021), p. 110322. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2021.110322>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999121002175>.
- [114] Rong Wang and Raymond J Spiteri. “Linear instability of the fifth-order WENO method”. In: *SIAM Journal on Numerical Analysis* 45.5 (2007), pp. 1871–1901. DOI: 10.1137/050637868. URL: <https://epubs.siam.org/doi/abs/10.1137/050637868>.

-
-
- [115] Mohamed El Ouafa, Stephane Vincent, and Vincent Le Chenadec. “Monolithic solvers for incompressible two-phase flows at large density and viscosity ratios”. In: *Fluids* 6.1 (2021), p. 23. URL: <https://doi.org/10.3390/fluids6010023>.
- [116] Zixuan Yang, Min Lu, and Shizhao Wang. “A robust solver for incompressible high-Reynolds-number two-fluid flows with high density contrast”. In: *Journal of Computational Physics* (2021), p. 110474. URL: <https://doi.org/10.1016/j.jcp.2021.110474>.
- [117] Zheng Li, Cheng Liu, Decheng Wan, et al. “Numerical Simulations of Droplet Impact onto a Pool Surface”. In: *The 30th International Ocean and Polar Engineering Conference*. International Society of Offshore and Polar Engineers. 2020.
- [118] Yadong Zeng, Han Liu, Qiang Gao, Ann Almgren, Amneet Pal Singh Bhalla, and Lian Shen. “A consistent adaptive level set framework for incompressible two-phase flows with high density ratios and high Reynolds numbers”. In: *Journal of Computational Physics* 478 (2023), p. 111971. URL: <https://doi.org/10.1016/j.jcp.2023.111971>.
- [119] D. Fuster, T. Arrufat, M. Crialesi-Esposito, Y. Ling, L. Malan, S. Pal, R. Scardovelli, G. Tryggvason, and S. Zaleski. “A momentum-conserving, consistent, Volume-of-Fluid method for incompressible flow on staggered grids”. In: *Computers & Fluids* (2018). arXiv: 1811.12327. URL: <http://arxiv.org/abs/1811.12327>.
- [120] Sagar Pal, Daniel Fuster, and Stéphane Zaleski. *A novel momentum-conserving, mass-momentum consistent method for interfacial flows involving large density contrasts*. 2021. DOI: 10.48550/ARXIV.2101.04142. URL: <https://arxiv.org/abs/2101.04142>.
- [121] Cheng Liu, Ruoqing Gao, and Changhong Hu. “A consistent mass – momentum flux computation method for the simulation of plunging jet”. In: *Physics of Fluids* 34.3 (2022), p. 032114. DOI: 10.1063/5.0084894. URL: <https://doi.org/10.1063/5.0084894>.

-
-
- [122] Qiu Jin, Dominic Hudson, and W Geraint Price. “A Combined Volume of Fluid and Immersed Boundary Method for Modeling of Two-Phase Flows With High Density Ratio”. In: *Journal of Fluids Engineering* 144.3 (2022). URL: <https://doi.org/10.1115/1.4052242>.
- [123] Tomislav Marić and Jun Liu. *Inconsistencies in Unstructured Volume-of-Fluid Methods for Two-Phase Flows with High Density Ratios - Code*. Nov. 2023. DOI: 10.48328/tudatalib-1276. URL: <https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/4018>.
- [124] Tomislav Marić and Jun Liu. *Inconsistencies in Unstructured Volume-of-Fluid Methods for Two-Phase Flows with High Density Ratios - Data*. Nov. 2023. DOI: 10.48328/tudatalib-1277. URL: <https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/4019>.
- [125] Tomislav Marić and Jun Liu. *Inconsistencies in Unstructured Volume-of-Fluid Methods for Two-Phase Flows with High Density Ratios - code repository*. 2023. URL: <https://github.com/tmaric/TwoPhaseFlow/tree/feature/density-ratio>.
- [126] J. Crank and P. Nicolson. “A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type”. In: *Mathematical Proceedings of the Cambridge Philosophical Society* 43.1 (1947), pp. 50–67. DOI: <https://doi.org/10.1017/S0305004100023197>.
- [127] SG Rubin and PK Khosla. “Higher-order numerical solutions using cubic splines”. In: *AIAA Journal* 14.7 (1976), pp. 851–858. DOI: <https://doi.org/10.2514/3.61427>.
- [128] Bram van Leer. “Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov’s method”. In: *Journal of Computational Physics* 32.1 (1979), pp. 101–136. ISSN: 0021-9991. DOI: [https://doi.org/10.1016/0021-9991\(79\)90145-1](https://doi.org/10.1016/0021-9991(79)90145-1). URL: <https://www.sciencedirect.com/science/article/pii/0021999179901451>.

-
-
- [129] B.P. Leonard. “A stable and accurate convective modelling procedure based on quadratic upstream interpolation”. In: *Computer Methods in Applied Mechanics and Engineering* 19.1 (1979), pp. 59–98. ISSN: 0045-7825. DOI: [https://doi.org/10.1016/0045-7825\(79\)90034-3](https://doi.org/10.1016/0045-7825(79)90034-3). URL: <https://www.sciencedirect.com/science/article/pii/0045782579900343>.
- [130] Philip L Roe. “Characteristic-based schemes for the Euler equations”. In: *Annual review of fluid mechanics* 18.1 (1986), pp. 337–365. DOI: <https://doi.org/10.1146/annurev.fl.18.010186.002005>.
- [131] Bram van Leer. “Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme”. In: *Journal of Computational Physics* 14.4 (1974), pp. 361–370. ISSN: 0021-9991. DOI: [https://doi.org/10.1016/0021-9991\(74\)90019-9](https://doi.org/10.1016/0021-9991(74)90019-9). URL: <https://www.sciencedirect.com/science/article/pii/0021999174900199>.
- [132] Arthur W Adamson, Alice Petry Gast, et al. *Physical chemistry of surfaces, 6th Edition*. Vol. 150. New York: Wiley, 1997. URL: <https://www.wiley.com/en-us/Physical+Chemistry+of+Surfaces%5C%2C+6th+Edition-p-9780471148739>.
- [133] Frédéric Couderc. “Développement d’un code de calcul pour la simulation d’écoulements de fluides non miscibles. Application à la désintégration assistée d’un jet liquide par un courant gazeux.” PhD thesis. Ecole nationale supérieure de l’aéronautique et de l’espace, 2007.
- [134] D. Zuzio and J.L. Estivalèzes. “An efficient block parallel AMR method for two phase interfacial flow simulations”. In: *Computers & Fluids* 44.1 (2011), pp. 339–357. ISSN: 0045-7930. DOI: <https://doi.org/10.1016/j.compfluid.2011.01.035>. URL: <https://www.sciencedirect.com/science/article/pii/S0045793011000429>.
- [135] Davide Zuzio, Jean-Luc Estivalèzes, and Bastien DiPierro. “An improved multiscale Eulerian – Lagrangian method for simulation of atomization process”. In: *Computers & Fluids* 176 (2018), pp. 285–301. ISSN: 0045-7930. DOI: <https://doi.org/10.1016/j.compfluid.2018.05.010>.

-
- 1016/j.compfluid.2016.12.018. URL: <https://www.sciencedirect.com/science/article/pii/S0045793016304017>.
- [136] Thibault Xavier, Davide Zuzio, Matthias Averseng, and J-L Estivalezes. “Toward direct numerical simulation of high speed droplet impact”. In: *Meccanica* 55 (2020), pp. 387–401.
- [137] M Darwish, L Mangani, and F Moukalled. “General fully implicit discretization of the diffusion term for the finite volume method”. In: *Numerical Heat Transfer, Part B: Fundamentals* 71.6 (2017), pp. 506–532. URL: <http://dx.doi.org/10.1080/10407790.2017.1330060>.
- [138] Suraj S Deshpande, Lakshman Anumolu, and Mario F Trujillo. “Evaluating the performance of the two-phase flow solver interFoam”. In: *Computational science & discovery* 5.1 (2012), p. 014016. URL: <https://doi.org/10.1088/1749-4699/5/1/014016>.
- [139] Stéphane Popinet. “Numerical models of surface tension”. In: *Annual Review of Fluid Mechanics* 50 (2018), pp. 49–75.
- [140] Yichen Huang and Bin Xie. “A generic balanced-force algorithm for finite volume method on polyhedral unstructured grids with non-orthogonality”. In: *Journal of Computational Physics* 479 (2023), p. 112010. URL: <https://doi.org/10.1016/j.jcp.2023.112010>.
- [141] Tomislav Marić and Jun Liu. *A residual-based non-orthogonality correction for force-balanced unstructured Volume-of-Fluid methods - codes repository*. 2023. URL: <https://github.com/tmaric/TwoPhaseFlow/tree/feature/non-orthogonality>.
- [142] Chae M Rhie and Wei-Liang Chow. “Numerical study of the turbulent flow past an airfoil with trailing edge separation”. In: *AIAA journal* 21.11 (1983), pp. 1525–1532. URL: <https://doi.org/10.2514/3.8284>.
- [143] Samir Muzafferija and David Gosman. “Finite-volume CFD procedure and adaptive error control strategy for grids of arbitrary topology”. en. In: *Journal of Computational Physics* 138.2 (Dec. 1997), pp. 766–787. ISSN: 00219991. URL: <https://doi.org/10.1006/jcph.1997.5853> (visited on 12/05/2023).

-
- [144] Ismet Demirdžić. “On the discretization of the diffusion term in finite-volume continuum mechanics”. In: *Numerical Heat Transfer, Part B: Fundamentals* 68.1 (2015), pp. 1–10. URL: <https://doi.org/10.1080/10407790.2014.985992>.
- [145] OpenCFD Ltd. *OpenFOAM-v2306*. 2022. URL: <https://develop.openfoam.com/Development/openfoam/-/tree/OpenFOAM-v2306> (visited on 12/06/2023).
- [146] Bin Xie, Peng Jin, and Feng Xiao. “An unstructured-grid numerical model for interfacial multiphase fluids based on multi-moment finite volume formulation and THINC method”. In: *International journal of multiphase flow* 89 (2017), pp. 375–398. URL: <https://doi.org/10.1080/10407790.2013.849996>.
- [147] Horacio J Aguerre, Cesar I Pairetti, Cesar M Venier, Santiago Márquez Damián, and Norberto M Nigro. “An oscillation-free flow solver based on flux reconstruction”. In: *Journal of Computational Physics* 365 (2018), pp. 135–148. URL: <https://doi.org/10.1016/j.jcp.2018.03.033>.
- [148] Ashwani Assam and Ganesh Natarajan. “A novel least squares finite volume scheme for discontinuous diffusion on unstructured meshes”. In: *Computers & Mathematics with Applications* 96 (2021), pp. 120–130. URL: <https://doi.org/10.1016/j.camwa.2021.05.013>.
- [149] Samir Muzaferija. “Adaptive finite volume method for flow prediction using unstructured meshes and multigrid approach”. PhD thesis. University of London UK, 1994. URL: <http://hdl.handle.net/10044/1/7728>.
- [150] Ismet Demirdžić and Samir Muzaferija. “Numerical method for coupled fluid flow, heat transfer and stress analysis using unstructured moving meshes with cells of arbitrary topology”. In: *Computer methods in applied mechanics and engineering* 125.1-4 (1995), pp. 235–255. URL: [https://doi.org/10.1016/0045-7825\(95\)00800-G](https://doi.org/10.1016/0045-7825(95)00800-G).
- [151] Hiroaki Nishikawa. “Beyond interface gradient: a general principle for constructing diffusion schemes”. In: *40th fluid dynamics conference and exhibit*. 2010, p. 5093. URL: <https://doi.org/10.2514/6.2010-5093>.

-
-
- [152] Hiroaki Nishikawa. “Robust and accurate viscous discretization via upwind scheme— I: Basic principle”. In: *Computers & Fluids* 49.1 (2011), pp. 62–86. URL: <https://doi.org/10.1016/j.compfluid.2011.04.014>.
- [153] Jian Wu and Philippe Traoré. “Similarity and comparison of three finite-volume methods for diffusive fluxes computation on nonorthogonal meshes”. In: *Numerical Heat Transfer, Part B: Fundamentals* 64.2 (2013), pp. 118–146. URL: <https://doi.org/10.1080/10407790.2013.784146>.
- [154] Alireza Jalali, Mahkame Sharbatdar, and Carl Ollivier-Gooch. “Accuracy analysis of unstructured finite volume discretization schemes for diffusive fluxes”. In: *Computers & Fluids* 101 (2014), pp. 220–232. URL: <https://doi.org/10.1016/j.compfluid.2014.06.008>.
- [155] PJ Linstrom, WG Mallard, et al. “NIST standard reference database number 69”. In: *NIST Chemistry WebBook* (2005), p. 20899. URL: <https://webbook.nist.gov/>.
- [156] VDI e. V. *VDI Heat Atlas*. Springer Berlin, Heidelberg, 2012. ISBN: 978-3-540-77876-9. URL: <https://doi.org/10.1007/978-3-540-77877-6>.
- [157] T Petrova and RB Dooley. “Revised release on surface tension of ordinary water substance”. In: *Proceedings of the International Association for the Properties of Water and Steam, Moscow, Russia* 63 (2014), pp. 23–27. URL: <http://www.iapws.org/relguide/Surf-H2O-2014.pdf>.
- [158] Christopher Greenshields. *OpenFOAM v11 User Guide*. London, UK: The OpenFOAM Foundation, 2023. URL: <https://doc.cfd.direct/openfoam/user-guide-v11>.
- [159] Henning Scheufler and Johan Roenby. “TwoPhaseFlow: An OpenFOAM based framework for development of two phase flow solvers”. In: *arXiv preprint arXiv:2103.00870* (2021).
- [160] Jay P Boris and David L Book. “Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works”. In: *Journal of computational physics* 11.1 (1973), pp. 38–69. URL: [https://doi.org/10.1016/0021-9991\(73\)90147-2](https://doi.org/10.1016/0021-9991(73)90147-2).

-
- [161] Steven T Zalesak. “Fully multidimensional flux-corrected transport algorithms for fluids”. In: *Journal of computational physics* 31.3 (1979), pp. 335–362. URL: [https://doi.org/10.1016/0021-9991\(79\)90051-2](https://doi.org/10.1016/0021-9991(79)90051-2).
- [162] Hanif Montazeri. “A Consistent Numerical Method for Simulating Interfacial Turbulent Flows”. Available at <https://hdl.handle.net/1807/24834>. PhD thesis. Toronto, Canada: University of Toronto, 2010.
- [163] Hanif Montazeri, Markus Bussmann, and Javad Mostaghimi. “Accurate implementation of forcing terms for two-phase flows into SIMPLE algorithm”. In: *International Journal of Multiphase Flow* 45 (2012), pp. 40–52. ISSN: 0301-9322. URL: <https://doi.org/10.1016/j.ijmultiphaseflow.2012.05.003>.
- [164] Hanif Montazeri and C.A. Ward. “A balanced-force algorithm for two-phase flows”. In: *Journal of Computational Physics* 257 (2014), pp. 645–669. ISSN: 0021-9991. URL: <https://doi.org/10.1016/j.jcp.2013.09.054>.
- [165] Kasper Møller. *Investigating an alternative discretization of the gravitational force when simulating interfacial flows using the interIsoFoam solver*. Tech. rep. In Proceedings of CFD with OpenSource Software, 2021, Edited by Nilsson. H., http://dx.doi.org/10.17196/OS_CFD#YEAR_2021. Roskilde University, 2022.
- [166] H. Jasak and H. G. Weller. “Application of the finite volume method and unstructured meshes to linear elasticity”. en. In: *International Journal for Numerical Methods in Engineering* 48.2 (May 2000), pp. 267–287. ISSN: 0029-5981, 1097-0207. URL: [https://onlinelibrary.wiley.com/doi/10.1002/\(SICI\)1097-0207\(20000520\)48:2%3C267::AID-NME884%3E3.0.CO;2-Q](https://onlinelibrary.wiley.com/doi/10.1002/(SICI)1097-0207(20000520)48:2%3C267::AID-NME884%3E3.0.CO;2-Q) (visited on 11/07/2022).
- [167] Suhas Patankar. *Numerical heat transfer and fluid flow*. CRC press, 1980.
- [168] Jian Luo, XY Hu, and Nikolaus A Adams. “A conservative sharp interface method for incompressible multiphase flows”. In: *Journal of Computational Physics* 284 (2015), pp. 547–565. URL: <https://doi.org/10.1016/j.jcp.2014.12.044>.

-
-
- [169] Y Deubelbeiss and BJP Kaus. “Comparison of Eulerian and Lagrangian numerical techniques for the Stokes equations in the presence of strongly varying viscosity”. In: *Physics of the Earth and Planetary Interiors* 171.1-4 (2008), pp. 92–111. URL: <https://doi.org/10.1016/j.pepi.2008.06.023>.
- [170] Adrian V Coward, Yuriko Y Renardy, Michael Renardy, and John R Richards. “Temporal evolution of periodic disturbances in two-layer Couette flow”. In: *Journal of Computational Physics* 132.2 (1997), pp. 346–361. URL: <https://doi.org/10.1006/jcph.1996.5640>.
- [171] Henrik Rusche. “Computational fluid dynamics of dispersed two-phase flows at high phase fractions”. PhD thesis. Imperial College London (University of London), 2003. URL: <https://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.402087>.
- [172] Bruno Lafaurie, Carlo Nardone, Ruben Scardovelli, Stéphane Zaleski, and Gianluigi Zanetti. “Modelling Merging and Fragmentation in Multiphase Flows with SURFER”. In: *Journal of Computational Physics* 113.1 (1994), pp. 134–147. ISSN: 0021-9991. DOI: <https://doi.org/10.1006/jcph.1994.1123>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999184711235>.

WORK EXPERIENCE

- 04/2021-09/2024 **Research associate / PhD. Student**
Mathematical Modeling and Analysis (MMA), TU Darmstadt
- Magna cum laude (1.0)
 - Research areas: Unstructured finite volume methods for two-phase flows with high density ratios

EDUCATION

- 04/2017-03/2021 **Master program**
Mechanical and Process Engineering, TU Darmstadt
- Final score: 1.91
 - Study areas: fluid dynamics, numerical computation, CFD
- 09/2012-07/2016 **Bachelor program**
Mechanical Engineering and Automation, Beijing University of Posts & Telecommunication, Beijing, China
- GPA: 84/ 100 (Top 10%)
 - Study areas: control, production & automation technology
- 09/2009-06/2012 Xiangyang No.4 Middle School, Hubei, China

PROJECT EXPERIENCES

- 08/2020-02/2021 **Mater thesis: Unstructured Level Set / Front Tracking method for two-phase flow with high density ratios (Note: 1.0)**
Darmstadt, Germany
- Implement a new consistent method to improve performance of existent OpenFOAM solver `lentFoam` in dealing with high density ratios (>1000)
- 01/2020-06/2020 **Working student by Outotec GmbH & Co. KG**
Oberursel, Frankfurt am Main, Germany
- CFD simulation with OpenFOAM: cleaning process of pigging tool, mixing process of agitator, modifying the design of back pressure vessel to reduce the dead zone
- 03/2020-08/2020 **HIWI job by MMA, TU Darmstadt: Algorithmische Geometrie für Mehrphasenströmungen in OpenFOAM**
Darmstadt, Germany
- Verifying and validating the newly developed solver `lentfoam` in OpenFOAM.

Conferences Contributions

“A numerically consistent unstructured Volume-of-Fluid discretization for the two-phase momentum convection with high density-ratios”, The 17th OpenFOAM Workshop, Ca Computers and Mathematics with Applicationsmbridge, UK, Jul. 11-14 2022

“A numerically consistent unstructured Volume-of-Fluid discretization for the two-phase momentum convection with high density-ratios”, SIAM Conference on Computational Science and Engineering (CSE23), Amsterdam, The Netherlands, Feb. 26-Mar. 3, 2023

“A numerically consistent unstructured Volume-of-Fluid discretization for the two-phase momentum convection with high density-ratios”, 93rd Annual Meeting of the International Association of Applied Mathematics and Mechanics, Dresden, Germany, May 30–Jun. 2, 2023

Publications

J. Liu, T. Tolle, D. Bothe, T. Marić. “An unstructured finite-volume Level Set / Front Tracking method for two-phase flows with large density-ratios”, *Journal of Computational Physics*, 2023, 112426, ISSN 0021-9991, <https://doi.org/10.1016/j.jcp.2023.112426>.

J. Liu, T. Tolle, D. Zuzio, J.L. Estivalezes, T. Marić. “Inconsistencies in Unstructured Geometric Volume-of-Fluid Methods for Two-Phase Flows with High Density Ratios.” *Computers & Fluids*, 2024, 106375, ISSN 0045-7930, <https://doi.org/10.1016/j.compfluid.2024.106375>.

J. Liu, T. Tolle, T. Marić. “A residual-based non-orthogonality correction for force-balanced unstructured Volume-of-Fluid methods” *Computers and Mathematics with Applications - Under Review*, Jun. 2024

G.L. Chai, **J. Liu**, D. Bothe, T. Tolle, J.W. Su, T. Marić. “Curvature approximation for the unstructured geometrical Volume-of-Fluid method” - *In Preparation*, Jun, 2024