

A unified multi-scale method for simulating two-phase immiscible flows

JOEL WRETBORN, Wētā FX and University of Waterloo, Sweden

ALEXEY STOMAKHIN, Wētā FX, USA

CHRISTOPHER BATTY, University of Waterloo, Canada



Fig. 1. *Waterfall*. Water plunges 400 m from a massive waterfall, transitioning from a smooth sheet near the outlet to fine spray as it is torn apart by the surrounding air. Our method captures both flow regimes using a single, unified representation. Left to right: simulation snapshots, with individual particles rendered using a diffuse shader ©Wētā FX. Top right: reference image from Yosemite falls, ©flyingdawnmarie.

We present a novel method for simulating large-scale, high density-ratio two-phase immiscible flows. Our approach adopts a two-fluid mixture formulation discretized on a shared Eulerian grid, in which phase fractions are represented using Lagrangian material points. We carefully adapt established closure models to this hybrid framework to capture the exchange of forces between phases, enabling the smooth transition between stratified (grid-resolved) and dispersed (sub-grid) flow regimes without recourse to secondary solvers. The resulting volumetric coupling forces are jointly integrated in a pressure projection, yielding a strongly coupled scheme that remains stable under large time steps and in highly dynamic scenarios. By combining these features, our method provides a unified, multi-scale discretization capable of capturing immiscible two-phase flows covering a large

range of spatial scales in a single simulation. We illustrate our method's capabilities on a diverse set of multi-scale scenarios, including a massive waterfall, a waterbombing airplane, large underwater bubbles, a geyser, and crashing ocean waves.

CCS Concepts: • **Computing methodologies** → **Physical simulation**.

ACM Reference Format:

Joel Wretborn, Alexey Stomakhin, and Christopher Batty. 2026. A unified multi-scale method for simulating two-phase immiscible flows. *ACM Trans. Graph.* 45, 6, Article 247 (December 2026), 20 pages. <https://doi.org/10.1145/3842566>

1 Introduction

Applications of liquid simulation in visual effects often require capturing fluid behavior across a vast range of scales within the same simulation. Consider, for example, a large waterfall descending a few hundred meters (Fig. 1). As water initially flows up to and over the ledge the surface is relatively smooth and may be adequately resolved with moderate discrete resolutions. However, as the falling water accelerates downward, air begins to rip chunks of water into successively smaller and smaller pieces, until they become tiny droplets tossed about by turbulent airflow.

Authors' Contact Information: Joel Wretborn, joel@wbn.se, Wētā FX and University of Waterloo, Stockholm, Sweden; Alexey Stomakhin, st.alexey@gmail.com, Wētā FX, Mililani, USA; Christopher Batty, christopher.batty@uwaterloo.ca, University of Waterloo, Waterloo, Canada.



This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

© 2026 Copyright held by the owner/author(s).

ACM 1557-7368/2026/12-ART247

<https://doi.org/10.1145/3842566>

The dominant solution strategy for simulating such two-phase immiscible flows (i.e., fluids separated by distinct interfaces) can be classified as *interface-resolving* methods, in which the geometry of the interface is fully described at or above the scale of the discrete elements [Bridson 2015]. Since surface detail is fundamentally limited by the underlying discretization size, such methods capture fine-scale detail by exploiting sparsity and spatial adaptivity [Ando and Batty 2020; Braun et al. 2025; Ferstl et al. 2016]. Even so, capturing the full range of dynamics for our waterfall scenario is infeasible: (sub-)millimeter water droplets can become uniformly dispersed throughout large regions of the air, spanning several orders of magnitude in spatial scale. The necessary number of degrees of freedom becomes prohibitively large, while limits on screen-space resolution ensure that viewers cannot perceive the fine-scale interfacial details.

A popular alternative is to augment interface-resolving methods with additional solvers to capture droplets and spray, either synchronously or in secondary passes [Braun et al. 2025; Lesser et al. 2022; Losasso et al. 2008; Stomakhin et al. 2023]. These methods have produced impressive visual effects, but because each component discretization uses a different set of fundamental equations, the fluid lacks a consistent unified physical model across scales. These discrepancies can thus give rise to visual artifacts, such as sudden “pops” when discrete elements transition between states, loss or gain of conserved quantities like mass and momentum, or visually inconsistent motion where the solvers overlap or interact.

Interface-averaging methods present a fundamentally different and comparatively under-explored approach. Assuming that the two phases are sufficiently intermingled that no grid-resolved interface exists, they rely on sub-grid modeling of the geometry and physics to account for under-resolved interfaces and thereby close the system of governing equations. These have been shown to be highly effective for immiscible mixtures, like spray in air or clouds of submerged bubbles, and two typical modeling strategies have been pursued: homogenized fluids [Kim et al. 2010; Patkar et al. 2013], where both phases share a single velocity field, and two-fluid models [Nielsen and Østerby 2013; Wretborn et al. 2022, 2025], where the phases instead are represented with their own sets of conservation laws and exchange volumetric forces. Traditionally, these methods were developed to target scenarios where the volume ratio of droplets or bubbles is low relative to the surrounding fluid.

Recently, several authors have proposed the design of *multi-scale* fluids solvers, which aim to produce both stratified (resolved) and dispersed (under-resolved) fluid features simultaneously [Frederix et al. 2021; Meller et al. 2021; Wretborn et al. 2025; Štrubelj and Tiselj 2011]. These approaches all leverage two-fluid models—i.e., they are all instances of interface-averaging methods—where they define appropriate sub-grid *closure* models to capture the respective regime. At this point, however, no existing method (interface-averaging or interface-resolving) has been shown to capture visually convincing large-scale stratified and dispersed flows simultaneously, using a single discretization.

The goal of the present work is therefore to develop a unified, multi-scale, two-phase fluid model capable of capturing both stratified and dispersed flow features at high density ratios, including smooth transitions between the two regimes. Our approach is inspired by two-fluid models proposed in the engineering literature

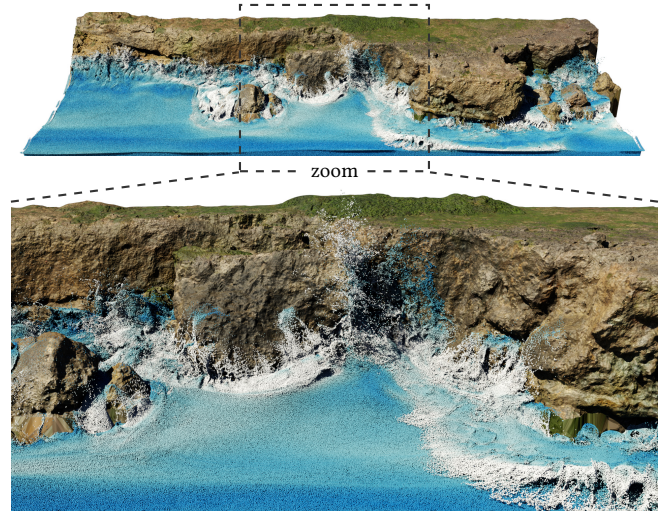


Fig. 2. *Coastal cliffs*. A 400 m coastline is hit by large waves. Simulated particles are rendered as spheres colored by speed. Our method simultaneously captures stratified flows as waves approach the cliffs and dispersed features as droplets are ejected tens of meters into the air. All particles have identical size. ©Wētā FX

[Frederix et al. 2021; Meller et al. 2021; Štrubelj and Tiselj 2011], but tailored to the demands of high-fidelity industrial visual effects. We achieve this by using a two-fluid formulation in which both phases are discretized on a shared Eulerian grid, and phase fractions are constructed by rasterizing massful particles. The principal challenges are threefold: (i) modifying Particle-In-Cell (PIC) to reduce its strong dependence on the relative size of particles and grid cells, which hampers its use in a sub-grid setting; (ii) designing appropriate terms for a closure model to capture the exchange of forces between the mixed but immiscible fluids, in both stratified and dispersed flow regimes; (iii) and stably incorporating these forces into the pressure projection for enforcing incompressibility. To this end we make the following key contributions:

- An adaptation of the Generalized Interpolation Material Point (GIMP; [Bardenhagen et al. 2004]) method to an incompressible FLIP discretization, allowing for a massful notion of particles by decoupling particle sizes from the underlying grid resolution.
- A hybrid particle-aware closure model, combining a linear closure term designed to model stratified flows [Štrubelj and Tiselj 2011] with a nonlinear sub-grid drag model for dispersed flows [Wretborn et al. 2022].
- An additional simplified virtual mass closure term that achieves stable phase coupling at high density ratios.
- A general two-field pressure projection algorithm, capable of strongly coupling the two-fluid model with any volumetric closure term proportional to the local relative phase velocity.

We begin by giving a high-level overview of our method (§4), followed by the discretization (§5) and full method description (§6). In §7, we show how to leverage our framework to adaptively refine the particle size or grid resolution. Then, in §8 we demonstrate the capabilities of our approach across a range of challenging scenarios.

Table 1. Categorization of multiphase flow methods.

Method	Examples	Interface treatment	Velocity fields	Regime
Particles with reconstructed level set	[Zhu and Bridson 2005]	sharp	single	stratified
	[Boyd and Bridson 2012]	sharp	single	stratified
	[Jiang et al. 2015]	sharp	single	stratified
Level set	[Fedkiw et al. 1999]	sharp	single	stratified
	[Losasso et al. 2008]	sharp	single	stratified
Volume-of-Fluid	[Popinet 2009]	sharp	single	stratified
	[Karnakov et al. 2022]	sharp	single	stratified
Phase-field	[Li et al. 2024]	diffuse	single	homogenized + stratified
Particles with reconstructed phase-indicator	[Kim et al. 2010]	diffuse	single	homogenized
	[Braun et al. 2025]	diffuse	single	stratified
Two-fluid (Eulerian-Lagrangian)	[Wretborn et al. 2022]	diffuse	multiple	dispersed
	[Wretborn et al. 2025]	diffuse	multiple	dispersed + stratified
Two-fluid (Eulerian-Eulerian)	[Nielsen and Østerby 2013]	diffuse	multiple	dispersed
	[Meller et al. 2021]	diffuse	multiple	dispersed + stratified
	[Frederix et al. 2021]	diffuse	multiple	dispersed + stratified
Two-fluid (Eulerian-FLIP)	Ours	diffuse	multiple	dispersed + stratified

2 Related work

We focus on immiscible two-phase flows, mainly targeting air-water dynamics. To structure the discussion, we categorize prior work along three axes: interface representation, the number of velocity fields, and intended flow regime. Interfaces may be represented sharply (as explicit geometry) or diffusely (as continuous fields). We distinguish between methods with a single (but potentially discontinuous) velocity field and multi-velocity formulations. We also distinguish between different flow regimes. With *stratified* flows, we refer to multiphase configurations where interfaces can be resolved by the underlying discretization. Stratified flows can be contrasted with sub-grid configurations—i.e., where interface detail cannot be discretely resolved—which we further divide into *dispersed* flows, which allow inter-phase slip¹ and *homogenized* flows, which assume phases share a common velocity field. Thus, the distinction between stratified and sub-grid regimes is governed by resolution, while the distinction between dispersed and homogenized flows is governed by the interface-averaging model used (slip versus no-slip).

Table 1 summarizes representative methods according to this categorization. We have excluded hybrid discretizations [Lesser et al. 2022; Patkar et al. 2013; Stomakhin et al. 2023] from the list above, since they achieve multiple flow regimes only by combining distinct discretizations. While the methods of Losasso et al. [2008] and Braun et al. [2025] are also hybrids, we categorize them according to their primary discretization (level set and particle phase-indicator, respectively).

Classical approaches such as FLIP, level set, and VOF methods employ sharp interface representations together with a single velocity field, and are therefore primarily suited for stratified flows. The success of these approaches in graphics has largely focused development on small-scale, surface-tension-dominated effects, where

¹While tangential(-only) interface slip is common in interface-resolving models, here and throughout this document we use the term inter-phase slip to refer to combined *tangential* and *normal* slip, which is necessary for dispersed models.

interfaces remain well resolved, such as glugging [Boyd and Bridson 2012; Goldade et al. 2020] and rising bubbles [Patkar et al. 2013]. However, recent work [Braun et al. 2025; Lesser et al. 2022; Stomakhin et al. 2023] has emphasized the importance of explicitly modeling air-water coupling in large-scale scenarios, such as our waterfall example. Despite the low density of air, the inertial effects arising from fast-moving water interacting with the surrounding air are essential to the resulting dynamics.

Phase-field approaches replace the sharp interface with a continuous representation but otherwise retain a single velocity field, resulting in homogenized models without inter-phase slip. For example, the model proposed by Li et al. [2024] allows for phases to unmix, but only through diffusion rather than relative phase flow.

Phase-indicator methods use particles to determine phase membership and rasterize this information—using, e.g., volume fractions [Kim et al. 2010] or sharpened density fields [Braun et al. 2025]—to represent a variable-density fluid. Unlike phase-field methods, particle phase-indicator methods do not directly evolve a phase field, instead relying on particle-based advection followed by reconstruction of a phase-indicator. While successful for both homogenized and stratified two-phase dynamics, these methods require additional models to approximate dispersed effects. For example, although Braun et al. [2025] showcase the benefits of a phase-indicator function, they retain the use of secondary Lagrangian particles to mimic dispersed spray [Braun et al. 2025, §7].

Two-fluid (mixture) models have historically seen limited use in computer graphics and, when adopted, have often focused on miscible fluids (e.g., [Ren et al. 2014]). An early example of immiscible fluids is the Eulerian-Eulerian model of Nielsen and Østerby [2013] for spray dynamics. While this approach effectively captures dispersed features, its semi-implicit coupling and fully Eulerian discretization are not intended to capture stratified flows with clear interfaces.

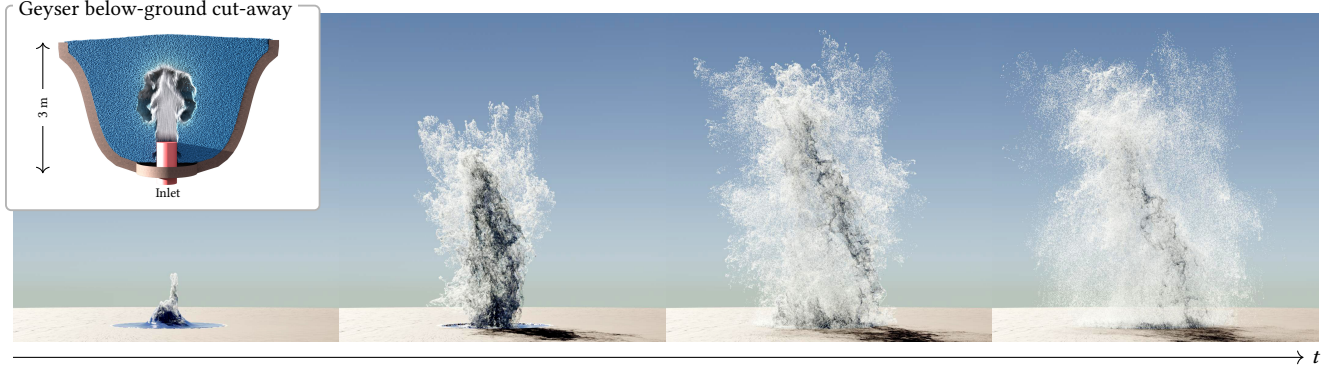


Fig. 3. *Geyser*. A geyser erupts, producing a large, expanding plume of fluid. While real-world geysers result from thermodynamic activity, we model the ejected fluid as a 10/90 water-air mixture to mimic the presence of water vapour. Top left: a cut-away showing particle geometry, colored by speed; the void just above the inlet corresponds to the emitted air phase. Left to right: a time series showing the plume expansion. Particles are rendered based on their dispersed classification (Eq. 25): dispersed particles are rasterized to a volumetric density field, others are surfaced as ellipsoids [Lesser et al. 2022]. Particles hitting the ground are removed from the simulation. ©Wētā FX

The closest method to ours is that of Wretborn et al. [2025], who investigated immersed bubbles and demonstrated that Eulerian-Lagrangian two-fluid models can be extended to capture both dispersed and stratified bubble dynamics. We largely adopt their particle representation, but differ in how particle motion is interpreted and updated: Wretborn et al. [2025] treat particles as physical Lagrangian bubbles, integrating their motion by interpolating pressure and drag forces from the ambient grid, whereas we view particles as samples of an Eulerian phase velocity field and update them using FLIP (§6.6). As such, their formulation employs a weakly coupled interaction between phases, solved using a Newton iteration loop that introduces significant computational cost. This strategy works well when the dominant inertial phase is described by the Eulerian representation, but can become unstable in stratified scenarios at the high density ratios considered in this work (see Fig. 20). In addition, while our closure models (§6.3) rely in part on the sub-grid drag model used by Wretborn et al. [2025], we adapt it to better handle resolved interfaces, and supplement it with a simplified virtual mass term, improving the stability of high density-ratio flows.

To our knowledge, no prior two-fluid method in graphics addresses water-air interactions at the spatial scales considered in this work. Because this represents a nonstandard use of two-fluid models, after introducing the equations of motion, we begin by providing necessary context for our modeling choices in §4.

3 Governing equations

The incompressible Euler equations for a two-phase mixture can be written as

$$\phi^s \rho^s \frac{D^s \mathbf{u}^s}{Dt} = \phi^s \rho^s \mathbf{g} - \phi^s \nabla p + \phi^s \mathbf{f}^\sigma + \mathbf{f}^{a \rightarrow s} \quad (1)$$

$$\phi^a \rho^a \frac{D^a \mathbf{u}^a}{Dt} = \phi^a \rho^a \mathbf{g} - \phi^a \nabla p + \phi^a \mathbf{f}^\sigma + \mathbf{f}^{s \rightarrow a} \quad (2)$$

$$\nabla \cdot (\phi^s \mathbf{u}^s + \phi^a \mathbf{u}^a) = 0, \quad (3)$$

$$\phi^s + \phi^a = 1. \quad (4)$$

Each phase is indicated by a superscript s (“submerged”) or a (“ambient”), and described by a phase fraction $\phi \in [0, 1]$, density ρ , and velocity $\mathbf{u}$; $\frac{D^*}{Dt}$ is the corresponding material derivative. The two fluids share a pressure field p and surface tension force $\mathbf{f}^\sigma$, and exchange interfacial forces via $\mathbf{f}^{s \rightarrow a} = -\mathbf{f}^{a \rightarrow s}$; the external force (typically gravity) is $\mathbf{g}$. Equations 1-4 together with conservation laws for ϕ^s , ρ^s , and ρ^a form a closed system of equations [Wretborn et al. 2025, App. A].

4 Method philosophy

We typically “see” only one of the phases in our day-to-day experience of two-phase flows: on a day at sea, we notice ocean waves and how wind affects the water, but the air itself is invisible; underwater, we can see bubbles rising but not how the surrounding water moves. We therefore define the notion of a primary—or *submerged*—phase: it is the fluid we wish to visualize, and it is surrounded by another *ambient* fluid.

Motivated by this distinction, we treat the two phases asymmetrically. Both are discretized on a shared Eulerian grid, where forces are exchanged and incompressibility is enforced. However, we prioritize the submerged phase and track it with material particles. The submerged phase is represented on the grid by rasterizing these particles, while the ambient phase is defined implicitly as the complement. Besides not needing to render the ambient phase, we make this choice for two reasons. First, a single set of particles avoids the complexity of sub-grid mixing of two distinct particle species, which otherwise requires careful treatment [Boyd and Bridson 2012; Braun et al. 2025]. More importantly, particle initialization in sparse dynamic domains with open boundaries is a subtle problem, requiring sophisticated adaptations of techniques like the fluxed animated boundary [Stomakhin and Selle 2017] or the simulation window [Flynn et al. 2026]. At a high level, our solver (Alg. 1) resembles traditional (single- or multi-phase) PIC solvers [Boyd and Bridson 2012; Zhu and Bridson 2005]. However, our method differs fundamentally in both the geometric interpretation of particles and the treatment of phase interaction.

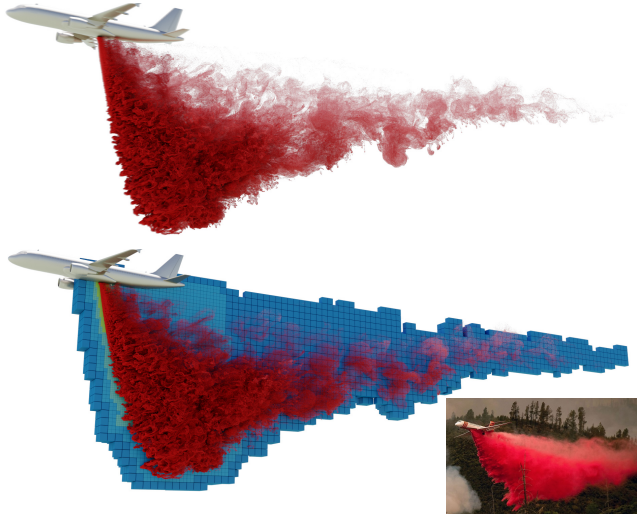


Fig. 4. *Waterbombing*. A total of 20 m^3 of liquid is released over 3 s from a fast-moving aircraft. Characteristic spike-like instabilities form on the water's leading edge as it penetrates the air. Simulated particles are rendered as red spheres at $3\times$ their original size. Top: rendered with motion blur. Bottom: adaptive domain buckets (§7.2), colored by size (four levels) ©Wētā FX. Bottom right: reference photograph ©Noah Berger / Getty Images.

PIC schemes rely on Lagrangian particles to represent the fluid material, but convert them to an Eulerian (implicit) representation to treat incompressibility and forces. Because of the dominance of interface-resolving methods, the most common choice of implicit field in graphics is a level set. This choice classifies the domain in a binary fashion into exterior or interior at the cell level. Particles must therefore have support on the order of the grid spacing, and the particle distribution is typically oversampled to avoid numerical gaps in the reconstructed field.² Thus, particle contributions to the level set depend on the local particle distribution, which prevents a consistent interpretation of particles as carriers of fixed material mass.

To avoid the issues of this grid-scale binary classification, we make use of particle basis functions Λ_q (§5.2) to reconstruct a smooth phase fraction field $\bar{\phi}^s$ describing the relative amount of submerged material at every point (§6.1). This description decouples the particle representation from the grid resolution by allowing partially filled voxels, and allows particle sizes and sampling density to be chosen independently from the Eulerian grid resolution.

However, a naive combination of this improved particle/grid representation with a standard single-velocity, variable-density pressure projection is insufficient for the class of flows we consider. Because such homogenized approaches assume that a single velocity field governs both phases, excessive numerical mixing is observed: particles gradually diffuse into the opposing phase, and stratified regions fail to remain separated (see Fig. 16).

The key to avoiding this diffusive behaviour is the multi-scale two-fluid coupling introduced in this work. Starting from the governing

²For example, for a regular PIC method with sphere-based implicit surface reconstruction, a particle must have a radius of at least $\sqrt{3/4}h$ to not disappear on the grid [Wretborn et al. 2025, §7].

equations of a two-phase mixture (Eqs. 1-3), our method adapts established two-fluid closure models [Frederix et al. 2021; Meller et al. 2021, 2023; Štrubelj and Tiselj 2011], previously only used in Eulerian-Eulerian discretizations, to blend between slip and no-slip interfacial conditions for dispersed and stratified flows, respectively. The local regime is determined from the submerged phase fraction and particle configuration. This is one of our primary contributions, and we outline the details in §6.3. The resulting momentum equations are then solved through a coupled pressure solve, as described in §6.4.

5 Discretization

Our method leverages both particles and a sparse, staggered Eulerian grid with voxel size h . In spatially adaptive domains (§7.2), h denotes the local voxel size unless specified otherwise. Grid cells are allocated around all submerged particles, with sufficient padding that no particles are involved in exterior boundary cells of the sparse domain. The construction of the sparse domain is outlined in App. A.

5.1 Notation

The set of cells is denoted by $\mathcal{C} \subset \mathbb{Z}^3$ with global index $i \in \mathcal{C}$. The set of staggered faces is denoted by $\mathcal{T}$ with global index $\tau \in \mathcal{T}$. We associate a control volume of size $V_i = V_\tau = h^3$ with each cell center and face center; we refer to either type of control volume as a voxel. Each face is additionally assigned the corresponding normal vector $\mathbf{e}_\tau$. The set of all particles is denoted by $\mathcal{P} \subset \mathbb{N}$, with index $q \in \mathcal{P}$. The simulation domain is denoted by Ω .

Unless otherwise stated, we refer to the discretization location of a quantity using a subscript i, τ, q for a cell, face, or particle, respectively. For example, V_q will refer to the volume of particle q , while $\mathbf{x}_i$ is the center position of cell i . We distinguish values

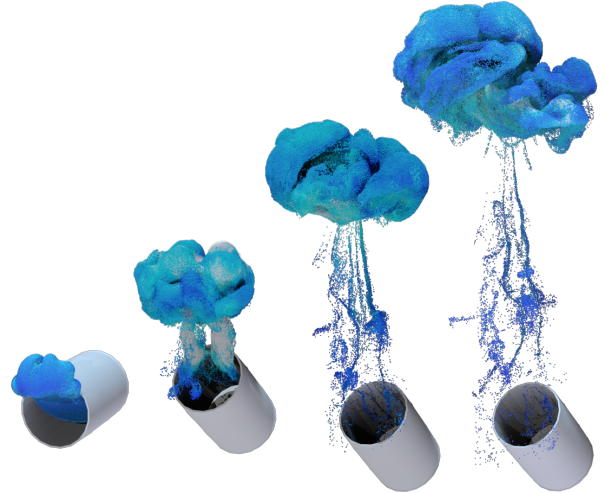


Fig. 5. *Bubble barrel*. An underwater barrel filled with 6 dm^3 air is overturned. Our method can reproduce stratified bubble features, similar to those demonstrated by Stomakhin et al. [2020] and Wretborn et al. [2025]. The simulation remains stable at a density ratio of 10:1000 with surface tension coefficient $\sigma = 0.072 \text{ N/m}$, whereas Wretborn et al. [2025] used a ratio of 1:10. ©Wētā FX

Table 2. Summary of notation used to describe our method. The discretized location (particle, cell, or face) of the quantities are provided where possible.

Notation	Units	Location	Meaning
r	m	q, i	particle radius
ρ^s	kg/m ³	q, i	submerged density
$\mathbf{u}^s$	m/s	q, τ	submerged velocity
$\phi^s \phi^a$	1	i	raw/corrected fraction
ρ^a	kg/m ³	i	ambient density
$\mathbf{u}^a$	m/s	τ	ambient velocity
p	N/m ²	i	pressure
f^σ	N/m ³	τ	surface tension force density
$f^{a \rightarrow s} = -f^{s \rightarrow a}$	N/m ³	τ	coupling force density
D	kg/m ³ s	τ	total closure term
D^L	kg/m ³ s	τ	linear drag closure term
D^{NL}	kg/m ³ s	τ	non-linear drag closure term
D^{VM}	kg/m ³ s	τ	virtual mass closure term
σ	N/m		surface tension coefficient
g	m/s ²		gravity
γ	s		closure relaxation time
ϕ^R	1		saturated fraction threshold
ϕ^{UR}	1		dispersed threshold
ψ^r	1		closure radius scale
ψ^{NL}	1		closure non-linear scale
h	m		voxel size
$\mathbf{x}$	m	q, i, τ	discrete element position
V	m ³	q, i, τ	discrete element volume
Λ	1		particle basis function
N	1		grid interpolation function
w	1		interpolation weights
Ω			simulation domain
D^a			simulation domain subset
$\mathcal{P}$			set of all particles
$\mathcal{C}$			set of all cells
$\mathcal{T}$			set of all faces
$\mathbf{e}$	1	τ	basis vector

at different discrete time steps with a subscript n . Omission of any subscript indicates we are referring to the underlying continuous field. Eulerian properties associated with a phase are disambiguated with a superscript: we use s for the submerged phase, and a for the ambient phase. For example, $\mathbf{u}_{n-1, \tau}^a$ is the ambient velocity at face τ at the previous time step $n-1$, $\mathbf{u}_q$ is the current velocity for particle q , and $\phi^s \equiv \phi^s(t, \mathbf{x})$ is the submerged fraction field.

A list of quantities used is provided in Tab. 2 together with the associated discrete elements.

5.2 Particle representation

We rasterize and interpolate particle quantities using the Generalized Interpolation Material Point (GIMP) method [Bardenhagen et al. 2004; Gao et al. 2017; Wretborn et al. 2025], which contrasts with most PIC-type methods in graphics, where implicit level set reconstruction combined with trilinear transfers are typical. This design choice is essential to decoupling the particle size from the grid resolution, and extends naturally to spatially adaptive grids.

Each particle $q \in \mathcal{P}$ is associated with a basis function $\Lambda_q : \Omega \rightarrow \mathbb{R}^+$ satisfying

$$\int_{\Omega} \Lambda_q(\mathbf{x}) d\mathbf{x} = V_q. \quad (5)$$

The weight w_{qi} between a particle and grid element (cell or face) $i \in \{\mathcal{C}, \mathcal{T}\}$ is defined as the convolution

$$w_{qi} = \frac{1}{V_q} \int_{\Omega} \Lambda_q(\mathbf{x}) N_i(\mathbf{x}) d\mathbf{x}, \quad (6)$$

for N_i an interpolation function defined per element which forms a partition of unity, i.e., $\sum_i N_i(\mathbf{x}) = 1, \forall \mathbf{x} \in \Omega$. For a full list of requirements of N_i , we refer the reader to [Gao et al. 2017, §3.3].

Using these weights, a scalar quantity $a \in \mathbb{R}^{|\mathcal{C}|}$ or vector quantity $\mathbf{b} \in \mathbb{R}^{|\mathcal{T}|}$ can be interpolated from the grid as

$$a_q \equiv a_i(\mathbf{x}_q) = \sum_{i \in \mathcal{C}} a_i w_{qi}, \quad \mathbf{b}_q \equiv \mathbf{b}_\tau(\mathbf{x}_q) = \sum_{\tau \in \mathcal{T}} \mathbf{b}_\tau w_{q\tau}. \quad (7)$$

We will use the $a_i(\mathbf{x})$ shorthand to explicitly denote the use of interpolation in formulas. Rasterization procedures using weights are outlined in §6.1.

Discussion. Because particle contributions are defined through the convolution in Eq. 6, particles may span multiple cells. For example, when using the box basis Λ^0 (Eq. 9), the resulting weights correspond to volume fractions, meaning that a particle can exactly fill a cell or face without over-filling. By contrast, standard PIC-style interpolation kernels would cause larger-than-a-voxel particles to cause artificial mass accumulation. We leverage this property in our method by optionally performing rasterization (§6.1) of particles onto a higher-resolution grid (2× the base resolution), which improves solver accuracy (Fig. 10).

5.2.1 Particle basis functions. We have implemented the following functions.

Delta function, Λ^δ . Let

$$\Lambda_q^\delta(\mathbf{x}) = V_q \delta_q(\mathbf{x} - \mathbf{x}_q), \quad (8)$$

for the Dirac-delta function $\delta(\mathbf{x})$. As pointed out by Gao et al. [2017], Eq. 8 combined with B-spline interpolants (Eq. 12) recovers regular PIC-style transfer kernels.

Box function, Λ^0 . We define $\text{cube}(V, \mathbf{x})$ to be a function that returns 1 if $\mathbf{x}$ is inside the cube with side length $\sqrt[3]{V}$ centered at the origin, and 0 otherwise. Then

$$\Lambda_q^0(\mathbf{x}) = \text{cube}(V_q, \mathbf{x} - \mathbf{x}_q). \quad (9)$$

Clamped box function, $\tilde{\Lambda}^0$. The box function Λ^0 can, when combined with a narrow grid interpolant and small particle sizes $V_q \ll h^3$, give rise to cell-crossing instabilities, resulting in particle jittering near cell boundaries. One solution is to clamp the support of Λ^0 to be at least one voxel in width. We thus define

$$\tilde{\Lambda}_q^0(\mathbf{x}) = \min[1, V_q/h^3] \text{cube}(\max[V_q, h^3], \mathbf{x} - \mathbf{x}_q). \quad (10)$$

5.2.2 Grid interpolants. The simplest interpolant N^0 is an indicator function

$$N_i^0(\mathbf{x}) = \text{cube}(V_i, \mathbf{x} - \mathbf{x}_i), \quad \forall i \in \{\mathcal{C}, \mathcal{T}\}. \quad (11)$$

Higher-order interpolants can be generated by successive convolution [Gao et al. 2017, §3.3] as

$$N_i^l(\mathbf{x}) = \int N^{l-1}(\mathbf{x}') N^0(\mathbf{x} - \mathbf{x}') d\mathbf{x}'. \quad (12)$$

We have used Eq. 12 to implement grid interpolants up to order $l = 3$. In §8 we demonstrate the impact of the choice of N^l .

5.3 Grid interpolation

In addition to particle-grid transfers, certain quantities stored on cells or faces must be interpolated to other grid locations. We define these interpolations within the same GIMP framework as above.

In particular, the zeroth-order interpolant (Eq. 11) satisfies the condition in Eq. 5 for all cells and faces. This allows us to define consistent weights $w_{\tau i}$ between any pair $i \in C$ and $\tau \in \mathcal{T}$, analogously to the particle-grid weights in Eq. 6. Using this construction, we denote interpolation of a scalar quantity $a \in \mathbb{R}$ from faces to cells as $a_{\tau}(x_i)$, and from cells to faces as $a_i(x_{\tau})$ (Eq. 7). All grid-to-grid interpolation in this work is performed using the zeroth-order interpolant N^0 , which, due to the convolution in Eq. 6, yields linear weights.

6 A two-fluid multi-scale simulator

We employ a standard operator splitting strategy [Stam 1999] to evolve Eqs. 1-3, separating advection from the evaluation of forces. The structure of our algorithm is laid out in Algorithm 1.

6.1 Rasterization

All submerged grid quantities must be reconstructed from particles (i.e., rasterized) on every simulation step.

Phase fraction. The submerged fraction $\bar{\phi}^s$ is discretized per cell $i \in C$ as

$$\bar{\phi}_i^s = \frac{1}{V_i} \sum_q V_q w_{qi}. \quad (13)$$

The fraction $\bar{\phi}^s$ is the ratio between rasterized particle volume and cell volume V_i . Because of the asymmetry between the submerged and ambient phases, the ambient fraction is represented only implicitly as the complement $1 - \bar{\phi}^s$ (Eq. 4). Near saturated cells, where $\bar{\phi}^s \rightarrow 1$, this complement is an unreliable measure of the small amount of ambient material present in the cell. Consequently, the raw fraction field is not used directly for simulation; instead, we apply the correction procedure outlined in §6.2.

Density. The submerged density ρ^s is discretized per cell $i \in C$ and rasterized as the volume-weighted mass

$$\rho_i^s = \frac{\sum_q w_{qi} \rho_q V_q}{\sum_q w_{qi} V_q}. \quad (14)$$

Velocity. The submerged velocity u^s is discretized per face $\tau \in \mathcal{T}$ and rasterized as the per-component mass-averaged momentum

$$u_{\tau}^s = \frac{\sum_q w_{q\tau} \rho_q V_q (u_q \cdot e_{\tau}) e_{\tau}}{\sum_q w_{q\tau} \rho_q V_q}. \quad (15)$$

Rasterization near boundaries. To avoid spurious gaps in rasterized submerged phase fraction and density near boundaries, we duplicate particles and reflect them across the boundary defined by the level set Φ^{solid} . Let $\mathcal{P}^{\text{solid}} \subset \mathcal{P}$ be the set of particles to be reflected, defined as

$$\mathcal{P}^{\text{solid}} = \{q \in \mathcal{P} : \Phi^{\text{solid}}(x_q) < \psi^{\text{solid}}\}, \quad (16)$$

Algorithm 1 Two-fluid multi-scale simulator

- 1: Advect (§6.7)
 - 2: Project particles out from boundaries (§6.7)
 - 3: Rasterize quantities (§6.1)
 - 4: Apply fraction correction (§6.2)
 - 5: Classify dispersed regions (Eqs. 26-27)
 - 6: Compute closure terms (§6.3)
 - 7: Apply explicit forces (§6.5)
 - 8: Perform pressure projection (§6.4)
 - 9: Apply FLIP velocity update (§6.6)
-

and use the index b to refer to the reflected particle corresponding to real particle q . The reflected position is defined as

$$x_b = x_q - 2\Phi^{\text{solid}}(x_q)\nabla\Phi^{\text{solid}}(x_q). \quad (17)$$

The reflection distance ψ^{solid} is chosen to ensure coverage of the boundary region,

$$\psi^{\text{solid}} = \frac{\sqrt{3}}{2} \left(\max_q (\text{supp}(\Lambda_q)) + \text{supp}(N) \right),$$

where $\text{supp}(\cdot)$ denotes support size.

The reflected particles $\mathcal{P}^{\text{solid}}$ are included when rasterizing fraction and density, i.e., the sums in Eqs. 13-14 are taken over both $\mathcal{P}$ and $\mathcal{P}^{\text{solid}}$.

Note the exclusion of reflected particles for velocity. While reflecting particle momentum across the boundary can improve the solver's accuracy, we found this to only be the case near smooth, well-behaved inputs. In practice, user-generated boundary geometry can have sharp kinks and rapidly varying conditions. Reflecting the momentum in such scenarios often caused instabilities. As a result we do not use reflected particles for velocity rasterization; that is, the sums over particles in Eq. 15 are taken over $\mathcal{P}$.

6.2 Fraction correction

The raw phase fractions $\bar{\phi}^s$ as defined in Eq. 13 may exhibit several non-physical artifacts due to the accumulation of small numerical errors. In particular, the particle distribution may become artificially clumped or artificially spread out relative to the physical flow. Since $\bar{\phi}^s$ is constructed from this particle distribution, such errors appear as local inconsistencies in the fraction field.

Artificial clumping leads to regions where $\bar{\phi}^s > 1$, indicating a local compression of material despite our incompressibility assumption. Artificial spreading gives rise to the opposite effect: in regions that should be fully saturated, the fraction field may contain small gaps or fluctuations. These variations are particularly problematic for $\bar{\phi}^s$ near 1, since small perturbations in the fraction field induce pressure variations which, for the large density ratios considered here, can drive significant flows. In effect, such spreading introduces small pockets of the ambient phase inside regions that should instead form a continuous domain of the submerged phase. Since the ambient phase is not tracked explicitly, these voids should be interpreted as numerical artifacts rather than physical structure.

The purpose of the correction proposed here is therefore to mitigate spurious flows caused by these numerical artifacts. For small $\bar{\phi}^s$, variations are physically meaningful and should be preserved

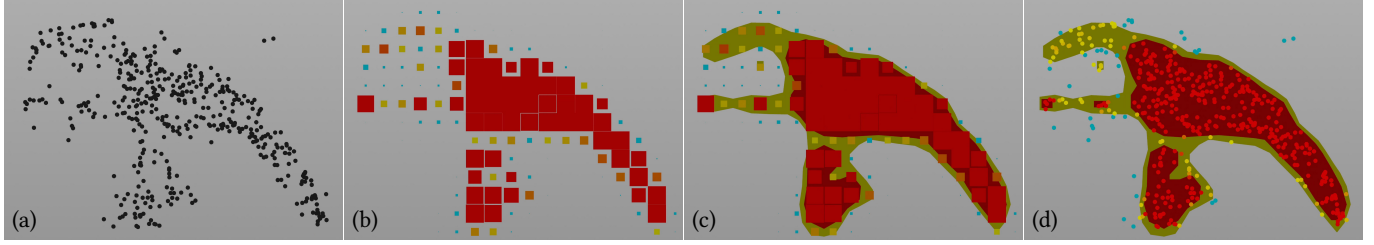


Fig. 6. *Fraction rasterization and flow regime classification.* (a) Particles (black circles) are (b) rasterized to produce the submerged fraction (colored squares, with size and color indicating the value in $[0, 1]$). (c) Two isocontours of ϕ^s are extracted for distinct purposes: $\phi^s = \phi^R$ (red), used for fraction correction (§6.2), and $\phi^s = \phi^{UR}$ (yellow), used for closure modeling (§6.3). (d) The fraction field is sampled at particle positions, $\phi^s(\mathbf{x}_q)$, to classify particles; dispersed particles $\mathcal{P}^{UR}$ (Eq. 25) are shown in teal. ©Wētā FX

(because the submerged phase is tracked explicitly). By contrast, small variations in $\tilde{\phi}^s$ near saturation are likely numerical artifacts, and the fraction field should remain contiguous.

We identify regions where numerical errors are likely as $\tilde{\phi}^s > \phi^R$, where $\phi^R \in [0, 1]$ is a user-defined threshold, and propose the following algorithm:

- (1) Convert the raw submerged fraction field to a distance field Φ of the ϕ^R isocontour, defined as $\Phi = \phi^R - \tilde{\phi}^s$.
- (2) Extrapolate one cell width into boundaries:

$$\Phi \leftarrow \min(\Phi, \max(\Phi - h, \Phi^{\text{solid}})). \quad (18)$$

- (3) Perform a morphological closing operation [Osher and Fedkiw 2002]:

$$\Phi \leftarrow \text{close}(\Phi, h). \quad (19)$$

- (4) Compute a corrected fraction field,

$$\phi^s = \begin{cases} \frac{1}{2} \left(1 + y + \frac{1}{\pi} \sin(\pi y) \right), & \Phi < 0 \\ \tilde{\phi}^s, & \Phi > 0 \end{cases} \quad (20)$$

$$\text{for the scaled distance } y = \text{clamp} \left(\phi^R - \frac{1}{2} - \frac{\Phi}{0.5h}, -1, 1 \right).$$

After each of the steps 1-3 in the above algorithm we additionally perform redistancing on the distance field Φ . The effect of Eq. 20 on the rasterized fractions is exemplified in Fig. 7. We demonstrate the effectiveness of this procedure in the ablation study of Fig. 15, and the effect of ϕ^R in the parameter study of Fig. 19b. Unless specified otherwise, all of our simulations use $\phi^R = 0.5$.

Extrapolation. Compared to $\tilde{\phi}^s$, the corrected fraction ϕ^s from Eq. 20 will have a greater number of nonzero fraction cells. These extra cells will need to be assigned measures of ρ^s and $\mathbf{u}^s$. For cell-centered quantities, we mark cells for which $\tilde{\phi}^s > 0$ as reliable and extrapolate until all cells with $\phi^s > 0$ have been written to. For face-centered quantities, we mark all faces for which a well-defined value has been written to as reliable (that is, for velocity $\mathbf{u}^s$, any face written to by Eq. 15) and extrapolate until any face touching a cell with $\phi^s > 0$ has been written to. Extrapolation is done using nearest-neighbour blending.

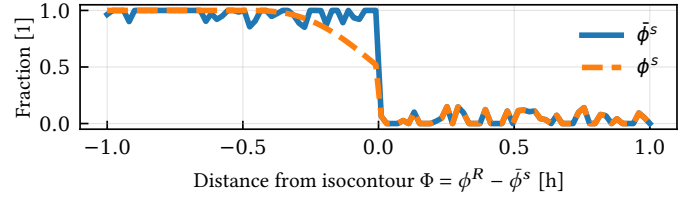


Fig. 7. *Fraction correction.* The fraction correction procedure (§6.2) removes spurious oscillations in the submerged phase fraction where $\tilde{\phi}^s > \phi^R$, leaving other regions unchanged. The value $\phi^R = 0.5$ was used for this plot.

6.3 Two-fluid closure terms

We define the total coupling force between the ambient and submerged phases proportional to the relative velocity $\Delta \mathbf{u} = \mathbf{u}^a - \mathbf{u}^s$,

$$\mathbf{f}^{a \rightarrow s} = -\mathbf{f}^{s \rightarrow a} = D(\phi, \rho, \mathbf{u}) \Delta \mathbf{u}, \quad (21)$$

where D is a spatially varying scalar function, referred to as a *closure term*, which may depend on the local fluid configuration. Since $\mathbf{f}^{a \rightarrow s}$ is a force density, the expected units of D are $\text{kg m}^{-3} \text{s}^{-1}$. The design of D is a modeling choice that determines the interaction of the two fluids. A larger D increases the coupling between the submerged and ambient phases, and as $D \rightarrow \infty$ the mixture converges toward a homogenized fluid: the two phases become “locked” together, effectively sharing the same velocity field.

6.3.1 Existing models. To employ a two-fluid model for stratified (resolved) flows, Štrubelj and Tiselj [2011] proposed a (linear) *drag closure term* proportional to the mixture density $\rho = \phi^s \rho^s + \phi^a \rho^a$,

$$D^{\text{Štrubelj}} = \phi^s \phi^a \rho \frac{1}{\gamma}, \quad (22)$$

where $\gamma \in \mathbb{R}^+$ is a relaxation time parameter. Štrubelj and Tiselj [2011] designed Eq. 22 to approximate a no-slip boundary condition between phases, and argued that γ must be “small enough” to ensure that the velocity of a phase dominates where the fraction of that field dominates. They use $\gamma = 10^{-2} \Delta t$, but more recent work uses relaxation times as small as $\gamma = 10^{-8} \Delta t$ [Meller et al. 2021, 2023].

However, Meller et al. [2023] pointed out that Eq. 22 overestimates the coupling force in under-resolved regions where a *slip* condition is more appropriate. They therefore proposed to blend between resolved and under-resolved closure models depending on a local

indicator function. Since we likewise wish to capture both extremes, we will take a similar approach.

To handle the dispersed (under-resolved) end of the spectrum, we will draw from the drag model of Wrethorn et al. [2022, Eq. 5], given by

$$D^{\text{Wrethorn}} = \phi^s \phi^a \left(\frac{6\pi\mu r}{V(r)} + \frac{\rho r^2 \pi}{2V(r)} \|\Delta \mathbf{u}\| \right), \quad (23)$$

where their original Lagrangian formulation is converted to an Eulerian force density in the same manner as Nielsen and Østerby [2013, §3.3]. Consisting of linear and non-linear terms, it has been successfully used to model under-resolved fluid-in-fluid interactions. We overload our notation slightly and use $V(r) = 4\pi r^3/3$ to express the volume of a sphere with radius r .

A *virtual mass* force term is often also used to stably capture high density ratio dispersed flows [Frederix et al. 2021; Meller et al. 2021, 2023]. We introduce the details of this term in the next section, and incorporate it into our formulation.

6.3.2 Our model. Inspired by the approaches above we define a combined closure model with three active terms: a linear drag model D^L , a quadratic drag model only active in under-resolved regions D^{NL} , and a simplified virtual mass model D^{VM} ,

$$D = D^L + D^{NL} + D^{VM}. \quad (24)$$

The rest of this section will define D^L , D^{NL} , and D^{VM} .

Dispersed classification. Similar to Meller et al. [2023], we want to define an indicator function for under-resolved interface regions on the grid. To this end, we will leverage our particle discretization together with the corrected submerged fraction ϕ^s (§6.2).

Define the set of dispersed particles by sampling the submerged fraction and comparing it to the user-defined threshold $\phi^{UR} \in [0, 1]$,

$$\mathcal{P}^{UR} = \{q \in \mathcal{P} : \phi_i^s(\mathbf{x}_q) < \phi^{UR}\}. \quad (25)$$

These are essentially isolated submerged particles far from dense regions of the submerged phase; see Fig. 6 for an illustration.

Our sub-grid closure models will rely on knowing both the local radius of these dispersed particles and which cells contain them. Therefore, we rasterize the normalized radius r_i^{UR} for dispersed particles to the grid,

$$r_i^{UR} = \frac{\sum_{q \in \mathcal{P}^{UR}} \sqrt[3]{3V_q/4\pi} w_{qi}}{\sum_{q \in \mathcal{P}^{UR}} w_{qi}}, \quad \forall i \in C. \quad (26)$$

and we consider a cell *dispersed* if $r_i^{UR} > 0$. Then, we define the (continuous) indicator function

$$\alpha_i = \begin{cases} \min(\phi_i^s/\phi^{UR}, 1), & \text{if } i \text{ is dispersed} \\ 1, & \text{otherwise} \end{cases} \quad \forall i \in C. \quad (27)$$

Equation 27 allows us to distinguish small submerged fractions ϕ^s in dispersed regions from those that are part of smoothly resolved (stratified) interfaces.

We demonstrate the effect of ϕ^{UR} in Fig. 19c; a typical value is $\phi^{UR} = 0.3$ [Frederix et al. 2021], which we use throughout this work. We emphasize that, while we leverage a binary classification

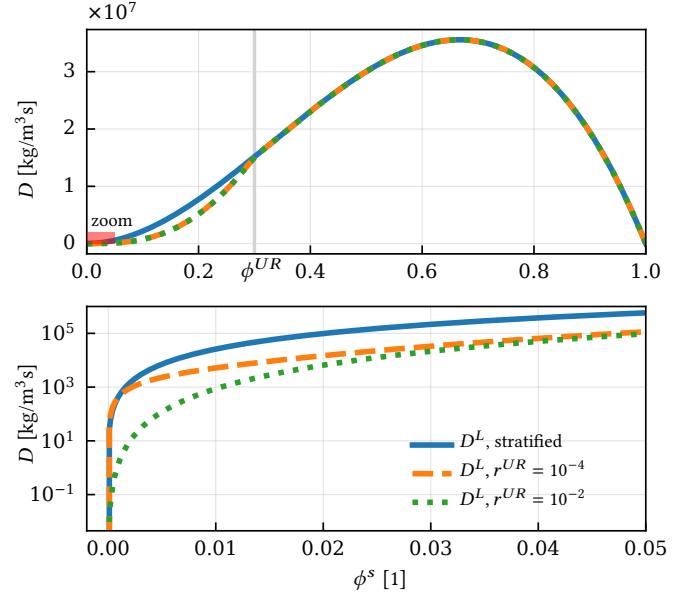


Fig. 8. *Linear closure model.* Linear drag closure (Eq. 29) for a 1:1000 mixture with relaxation time $\gamma = 10^{-4}$ (Eq. 22) and $\psi^r = 1$. Dispersed regions (dashed) are defined where the under-resolved radius is non-zero (Eq. 26); two choices of r^{UR} are shown. Stratified regions (blue, filled) correspond to locations where the under-resolved radius is undefined (i.e., no dispersed particles are nearby), in which case the model reduces exactly to [Štrubelj and Tiselj 2011]. Top: the full submerged fraction range. Bottom: a zoomed-in region near the origin (zoom box shown in red). Small differences in closure values lead to markedly different flow behavior; compare Fig. 9a vs. 9b. ©Wētā FX

mechanism in Eq. 25 and 27, these do not constitute hard switching of different models within our method. Instead, phase coupling is determined entirely by our closure model (Eq. 24), which is continuous and smoothly blends between dispersed and stratified flow regimes.

Linear drag. We want to leverage Eq. 22 in stratified regions and the linear term of the drag model of Eq. 23 in dispersed regions, with a smooth transition. We define an *effective resolved drag radius* $r^R \in \mathbb{R}^+$ per cell by matching the linear Stokes drag force per unit volume to the relaxation time,

$$\frac{6\pi\mu r_i^R}{V(r^R)} = \frac{\rho_i}{\gamma} \implies r_i^R = \sqrt{\frac{9\gamma\mu}{2\rho_i}},$$

where $\rho_i = \phi_i^s \rho_i^s + \phi_i^a \rho_i^a$ is the local density of the mixture. That is, the relaxation time in Eq. 22 needed for strong coupling of the two phases in resolved regions can equally well be viewed in terms of a (spatially varying) particle radius r_i^R .

Then, we use the indicator function α_i (Eq. 27) to linearly interpolate between the resolved and under-resolved radii (Eq. 26):

$$r_i = \alpha_i r_i^R + (1 - \alpha_i) \psi^r r_i^{UR}, \quad \forall i \in C. \quad (28)$$

The unitless scale $\psi^r \in \mathbb{R}^+$ is a modeling parameter that allows the user to adjust the effective particle size. This is useful since a simulated particle may represent an aggregate of unresolved material

(e.g., many smaller droplets), rather than a single physical entity. We demonstrate the effect of ψ^r in Figs. 9 and 19a.

Finally, the linear drag closure term is defined on faces as

$$D_{\tau}^L = \phi_i^s(\mathbf{x}_{\tau}) \phi_i^a(\mathbf{x}_{\tau}) \frac{6\pi\mu r_i(\mathbf{x}_{\tau})}{V(r_i(\mathbf{x}_{\tau}))}. \quad (29)$$

Note that ϕ_i^s , ϕ_i^a , and r_i in Eq. 29 are all defined on cells and must be interpolated to faces (§5.3). We visualize D^L for a few flow conditions in Fig. 8. Importantly, we recover the non-slip model by Štrubelj and Tiselj [2011] exactly in stratified flow regimes. Figure 15c demonstrates what happens if inter-phase coupling is too weak: stratified regions collapse, causing the submerged liquid to lose volume. We have used $\gamma = 10^{-4}\Delta t$ in all of our simulations.

Quadratic drag. In dispersed regimes we use the same quadratic drag closure as Wretborn et al. [2022] using the indicator function as a mask:

$$D_{\tau}^{NL} = \psi^{NL} (1 - \alpha_i(\mathbf{x}_{\tau})) \phi_i^s(\mathbf{x}_{\tau}) \phi_i^a(\mathbf{x}_{\tau}) \frac{\pi r_i(\mathbf{x}_{\tau})^2 \rho_i(\mathbf{x}_{\tau})}{2 V(r_i(\mathbf{x}_{\tau}))} \|\Delta \mathbf{u}\|_{\tau}. \quad (30)$$

In order to measure the relative velocity for a face we interpolate each phase velocity $\|\Delta \mathbf{u}\|_{\tau} = \|\mathbf{u}_{\tau}^a(\mathbf{x}_{\tau}) - \mathbf{u}_{\tau}^s(\mathbf{x}_{\tau})\|$ using §5.3. The user-defined parameter $\psi^{NL} \in \mathbb{R}^+$ controls the strength of the quadratic drag force.

Simplified virtual mass. Virtual mass, or “added mass”, is a force that accounts for the inertia of the surrounding fluid that must be accelerated together with an immersed object [Michaelides et al. 2022, §4.11.1]. While the drag force determines the terminal velocity, the virtual mass force limits the rate of acceleration. This effect becomes particularly important in high density-ratio flows, where it helps prevent unphysical acceleration and improves the stability of the dispersed regime. For dispersed two-fluid models it is common to compute this force directly as the relative acceleration [Frederix et al. 2021; Meller et al. 2021, 2023]

$$\mathbf{f}^{VM} = \phi^s \phi^a \rho^a \psi^{VM} \left(\frac{D^a \mathbf{u}^a}{Dt} - \frac{D^s \mathbf{u}^s}{Dt} \right) \quad (31)$$

where the ψ^{VM} depends on the geometry of the submerged object. The analytical value for a sphere is $\psi^{VM} = 0.5$, which we use throughout this work. Analogous to operator splitting, we propose to discretize Eq. 31 without the convective terms, i.e.,

$$\mathbf{f}^{VM} \approx \phi^s \phi^a \rho^a \psi^{VM} \left(\frac{\partial \mathbf{u}^a}{\partial t} - \frac{\partial \mathbf{u}^s}{\partial t} \right). \quad (32)$$

While this approximation neglects nonlinear transport and therefore underestimates the virtual mass force in turbulent regions, the simplified formulation makes the computation tractable and still captures the dominant inertial coupling between phases.

We discretize Eq. 32 with backwards Euler $\partial \mathbf{u}_{\tau}^{\star} / \partial t \approx (\mathbf{u}_{\tau, n+1}^{\star} - \mathbf{u}_{\tau, n}^{\star}) / \Delta t$, $\star \in \{s, a\}$, which naturally divides the force into an implicit and explicit component. The resulting force expression is linear in the relative phase velocity, $\Delta \mathbf{u}$, so we can write our virtual mass closure term as

$$D_{\tau}^{VM} = \frac{1}{\Delta t} \phi_i^s(\mathbf{x}_{\tau}) \phi_i^a(\mathbf{x}_{\tau}) \rho_i^a(\mathbf{x}_{\tau}) \psi^{VM}. \quad (33)$$

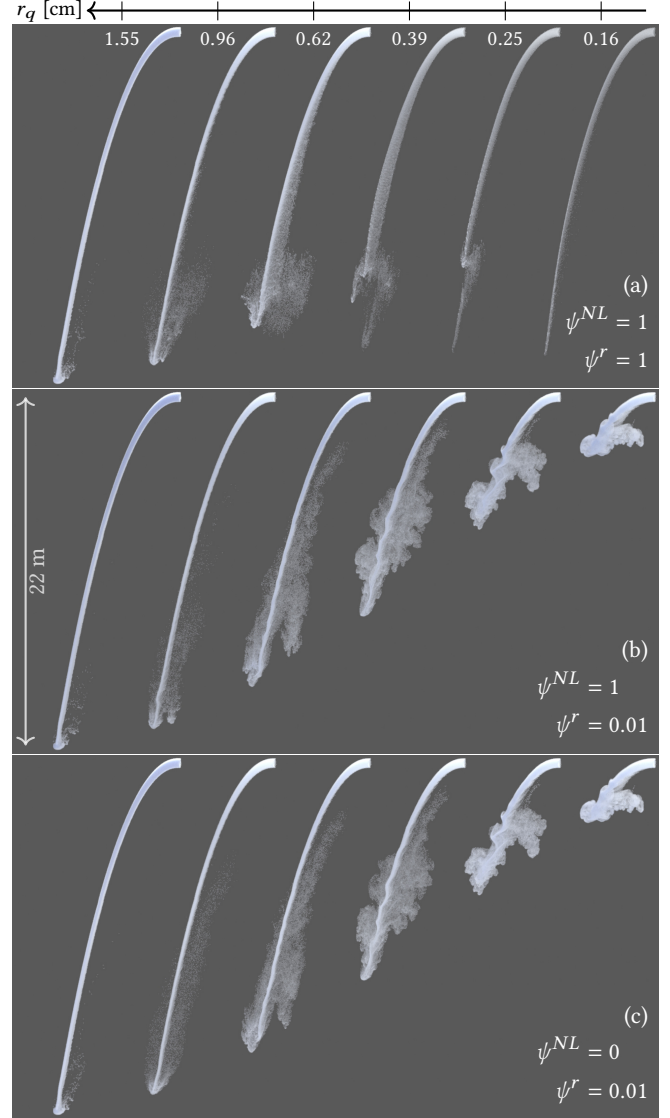


Fig. 9. *Effect of particle size.* A 1 m diameter sphere emits fluid at 5 m/s for varying particle sizes and closure parameters; particle counts and grid size are kept constant. The effective radius $\psi^r r_q$ (Eq. 28) affects the inter-phase drag-coupling, causing identical particles to exhibit behavior that is droplet-like (top) or fine spray-like (middle) in the dispersed regime. The nonlinear drag term subtly enhances turbulent breakup; here the trailing streams of spray are more dynamic and less smooth (cf. middle and bottom). ©Wētā FX

The implicit part will be integrated into the pressure solve together with the other two closure terms, D^L and D^{NL} , in the next section. We write the remaining explicit component as an explicit force,

$$\mathbf{f}_{\tau}^{VM} = D_{\tau}^{VM} (\mathbf{u}_{\tau}^a - \mathbf{u}_{\tau}^s), \quad (34)$$

to be integrated alongside gravity and surface tension in §6.5.

6.4 Two-fluid pressure solver

Pressure is a shared field acting on both the submerged and ambient phases. Following operator splitting, explicit forces are applied first, after which incompressibility of the mixture is enforced. This leads to a coupled elliptic system in which pressure acts as a Lagrange multiplier enforcing the divergence-free constraint (Eq. 3).

A key aspect of our formulation is that the closure force (Eq. 21) is proportional to the relative phase velocity, allowing it to be integrated implicitly as part of the coupled pressure solve in the stratified regime. In dispersed regions, due to the dependency on velocity in the nonlinear drag closure D^{NL} , the scheme is only semi-implicit. We have not observed any stability issues in practice.

Matrix notation. For some arbitrary quantity $Q_s \in \mathbb{R}$ defined for a discrete element s , we define a corresponding tensor $Q_S \in \mathbb{R}^{|S|}$ with $s \in S$. For example, $\mathbf{u}_{\mathcal{T}}^s \in \mathbb{R}^{|\mathcal{T}|}$ is the column vector of all submerged face velocities. We additionally define $\mathbb{I}_S$ as the $|S| \times |S|$ diagonal identity matrix, and the shorthand $[Q_S]$ for the function that diagonalizes a column vector.

Discretized equations of motion. Let $\mathbf{G} \in \mathbb{R}^{|\mathcal{T}| \times |\mathcal{C}|}$ and $\mathbf{G}^T \in \mathbb{R}^{|\mathcal{C}| \times |\mathcal{T}|}$ denote the discrete finite difference gradient and divergence operators on a staggered grid, respectively. In what follows, we omit a temporal subscript and write $\mathbf{u}_{\mathcal{T}}^s \equiv \mathbf{u}_{n+1,\mathcal{T}}^s$, and similarly for the ambient phase. The unknowns are the new velocities $\mathbf{u}_{\mathcal{T}}^s$ and $\mathbf{u}_{\mathcal{T}}^a$, which satisfy the divergence-free constraint; $\bar{\mathbf{u}}^s$ and $\bar{\mathbf{u}}^a$ are phase velocities after forces have been applied (§6.5).

An implicit temporal discretization of Eqs. 1-3, neglecting advection, yields

$$\begin{pmatrix} [m_{\mathcal{T}}^s + \Delta t D_{\mathcal{T}}] & -\Delta t [D_{\mathcal{T}}] & \Delta t [\phi_{\mathcal{T}}^s] \mathbf{G} \\ -\Delta t [D_{\mathcal{T}}] & [m_{\mathcal{T}}^a + \Delta t D_{\mathcal{T}}] & \Delta t [\phi_{\mathcal{T}}^a] \mathbf{G} \\ \mathbf{G}^T [\phi_{\mathcal{T}}^s] & \mathbf{G}^T [\phi_{\mathcal{T}}^a] & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\mathcal{T}}^s \\ \mathbf{u}_{\mathcal{T}}^a \\ p_C \end{pmatrix} = \begin{pmatrix} [m_{\mathcal{T}}^s] \bar{\mathbf{u}}_{\mathcal{T}}^s \\ [m_{\mathcal{T}}^a] \bar{\mathbf{u}}_{\mathcal{T}}^a \\ 0 \end{pmatrix}, \quad (35)$$

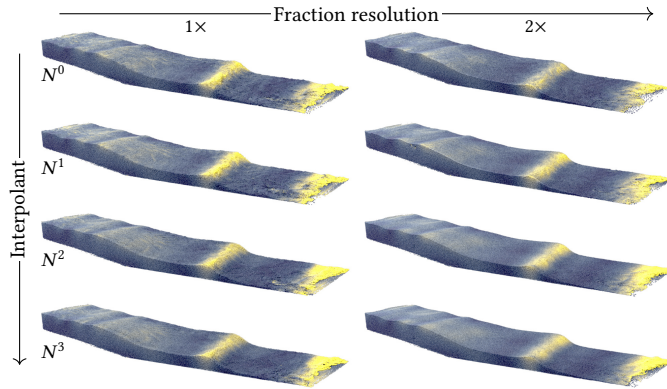


Fig. 10. *Beach wave.* We vary the interpolant and the resolution of the submerged fraction field for an otherwise identical simulation. A snapshot is taken after 10 s of simulation. Particles are colored by speed to emphasize local velocity differences. Grid artifacting (visualized as noise in the velocity field) can appear in stratified regions due to the absence of a sharp-interface treatment in our model; these issues are alleviated by increasing the interpolation order, or rasterizing ϕ^s to a finer grid. ©Wētā FX

where the closure force (Eq. 21) is discretized as

$$\mathbf{f}_{\mathcal{T}}^{a \rightarrow s} = -\mathbf{f}_{\mathcal{T}}^{s \rightarrow a} = [D_{\mathcal{T}}] \left(\mathbf{u}_{\mathcal{T}}^a - \mathbf{u}_{\mathcal{T}}^s \right),$$

and we use the fractional mass density $m_{\mathcal{T}}^{\star} = \phi_{\mathcal{T}}^{\star} \rho_{\mathcal{T}}^{\star}$, $\star \in \{s, a\}$. It is possible to block-diagonalize the two first rows of Eq. 35, following an analogous derivation to Fei et al. [2018, Supp. 1]; a derivation with our notation is provided in App. B. This yields

$$\begin{pmatrix} \mathbf{M}^s & 0 & \Delta t \mathbf{F}^s \mathbf{G} \\ 0 & \mathbf{M}^a & \Delta t \mathbf{F}^a \mathbf{G} \\ \mathbf{G}^T [\phi_{\mathcal{T}}^s] & \mathbf{G}^T [\phi_{\mathcal{T}}^a] & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\mathcal{T}}^s \\ \mathbf{u}_{\mathcal{T}}^a \\ p_C \end{pmatrix} = \begin{pmatrix} \mathbf{P}^s \bar{\mathbf{u}}_{\mathcal{T}}^s + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^a \\ \mathbf{P}^a \bar{\mathbf{u}}_{\mathcal{T}}^a + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^s \\ 0 \end{pmatrix} \quad (36)$$

with

$$\mathbf{M}^s = \Delta t [D_{\mathcal{T}}] \left([m_{\mathcal{T}}^a]^{-1} [m_{\mathcal{T}}^s] + \mathbb{I}_{\mathcal{T}} \right) + [m_{\mathcal{T}}^s], \quad (37)$$

$$\mathbf{F}^s = \Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} + [\phi_{\mathcal{T}}^s], \quad (38)$$

$$\mathbf{P}^s = \Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} [m_{\mathcal{T}}^s] + [m_{\mathcal{T}}^s]. \quad (39)$$

Since the two phases are symmetric, the ambient-phase versions of Eqs. 37-39 can be retrieved by swapping superscripts. From Eq. 36 we can eliminate the velocity variables and construct an equation involving only the pressure p_C :

$$\begin{aligned} & \underbrace{-\Delta t \mathbf{G}^T \left([\phi_{\mathcal{T}}^s] (\mathbf{M}^s)^{-1} \mathbf{F}^s + [\phi_{\mathcal{T}}^a] (\mathbf{M}^a)^{-1} \mathbf{F}^a \right) \mathbf{G} p_C}_{\text{mass matrix}} = \\ & \underbrace{-\mathbf{G}^T [\phi_{\mathcal{T}}^s] (\mathbf{M}^s)^{-1} \left(\mathbf{P}^s \bar{\mathbf{u}}_{\mathcal{T}}^s + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^a \right) +}_{\text{mass flux terms}} \\ & \underbrace{-\mathbf{G}^T [\phi_{\mathcal{T}}^a] (\mathbf{M}^a)^{-1} \left(\mathbf{P}^a \bar{\mathbf{u}}_{\mathcal{T}}^a + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^s \right)}_{\text{mass flux terms}}. \end{aligned} \quad (40)$$

Equation 40 defines a symmetric positive definite Poisson problem with variable coefficients and non-zero right-hand-side. We solve the resulting linear system using a preconditioned conjugate gradient (CG) solver with algebraic multigrid (AMG) preconditioner [Lesser et al. 2022]. Once p_C is solved for, we recover phase velocities by substitution into Eq. 36,

$$\mathbf{u}_{\mathcal{T}}^s = (\mathbf{M}^s)^{-1} \left(\mathbf{P}^s \bar{\mathbf{u}}_{\mathcal{T}}^s + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^a - \Delta t \mathbf{F}^s \mathbf{G} p_C \right), \quad (41)$$

$$\mathbf{u}_{\mathcal{T}}^a = (\mathbf{M}^a)^{-1} \left(\mathbf{P}^a \bar{\mathbf{u}}_{\mathcal{T}}^a + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^s - \Delta t \mathbf{F}^a \mathbf{G} p_C \right). \quad (42)$$

Removing singularities. All singularities in Eqs. 37-42 are removable singularities; that is, all terms are finite values in the limit $\phi^{\star} \rightarrow \{0, 1\}$. Numerically we avoid issues by only computing the first terms in Eqs. 37-39 when $\phi^s > 0$ and $\phi^a > 0$, and in Eqs. 40-42 by only computing terms when the corresponding fraction is positive (e.g., $(\mathbf{M}^s)^{-1}$ is only computed if $\phi^s > 0$).

Ambient border boundary conditions. Only the ambient phase is present at the exterior boundaries of the simulation domain Ω , because our sparse grid domain is constructed with a sufficient buffer of cells around all submerged particles (App. A). We use the Dirichlet hydrostatic pressure boundary condition defined by Wretborn et al. [2022, 2025] at domain boundaries in all simulations except the geyser (Fig. 3), where a gentle $\mathbf{u}_{\partial\Omega}^a = \mathbf{1} \mathbf{e}_x$ m/s breeze pushes the spray to the side for added visual interest.

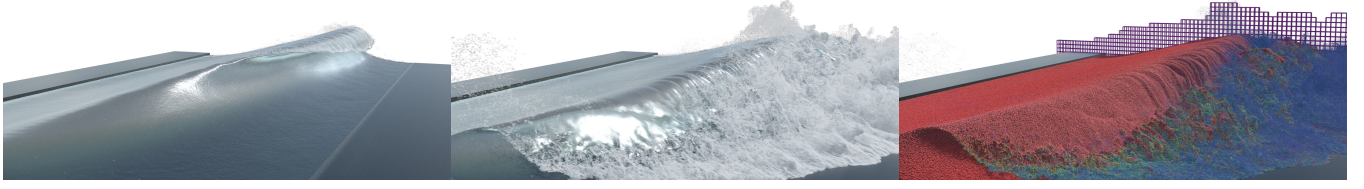


Fig. 11. *Barrel wave*. A traveling Stokes wave propagates toward a wall (not rendered), forcing it to break. Left and middle: two snapshots of the wave. Dispersed particles (Eq. 25) are rendered as mist via a fog volume, while remaining particles are converted to a level set using the method of Stomakhin et al. [2023] and rendered with a liquid shader. Right: a particle breakdown colored by size; hollow blocks represent the sparse domain data structure (App. A). As the wave steepens and breaks, a sharp air-water interface is maintained and transitions naturally into dispersed spray. The simulation uses the volumetric splitting strategy (§7.1). ©Wētā FX

Velocity extrapolation. To enable robust sampling of the velocity fields $\mathbf{u}_\tau^s$ and $\mathbf{u}_\tau^a$ we extrapolate the solutions of Eqs. 41-42 into velocity boundary conditions using nearest-neighbour blending.

In addition, we overwrite ambient velocities for faces $\tau \in \mathcal{T}$ where $\phi_i^a(\mathbf{x}_\tau) = 0$ such that $\mathbf{u}_\tau^a = \mathbf{u}_\tau^s$. This defines the ambient velocity in regions where it would otherwise be undefined, ensuring well-posed backtracing during ambient velocity advection (§6.7).

6.5 Application of explicit forces

To apply explicit forces, we evaluate

$$\bar{\mathbf{u}}_\tau^s = \mathbf{u}_\tau^s + \frac{\Delta t}{\rho_i^s} \left(\mathbf{g}(\mathbf{x}_\tau) + \mathbf{f}_\tau^\sigma + \frac{\mathbf{f}_\tau^{VM}}{\phi_i^s(\mathbf{x}_\tau)} \right), \quad (43)$$

$$\bar{\mathbf{u}}_\tau^a = \mathbf{u}_\tau^a + \frac{\Delta t}{\rho_i^a} \left(\mathbf{g}(\mathbf{x}_\tau) + \mathbf{f}_\tau^\sigma - \frac{\mathbf{f}_\tau^{VM}}{\phi_i^a(\mathbf{x}_\tau)} \right). \quad (44)$$

where $\bar{\mathbf{u}}$ denotes the newly updated velocity values. External forces $\mathbf{g} : \Omega \rightarrow \mathbb{R}^3$ are provided by the user and trilinearly interpolated to faces. The surface tension force $\mathbf{f}^\sigma$ is computed following Wretborn et al. [2025]. Note that division by phase fraction of the explicit virtual mass force term is not a true singularity at $\phi^* = 0$, since $\mathbf{f}_\tau^{VM} \propto \phi_i^s(\mathbf{x}_\tau)\phi_i^a(\mathbf{x}_\tau)$ from Eq. 33.

6.6 Velocity update

Particle velocities are updated using the FLIP scheme [Brackbill and Ruppel 1986; Zhu and Bridson 2005]

$$\mathbf{u}_{n+1,q} = v \left(\mathbf{u}_{n,q} + \Delta \mathbf{u}_{n+1,\tau}^s(\mathbf{x}_q) \right) + (1-v) \mathbf{u}_{n+1,\tau}^s(\mathbf{x}_q), \quad (45)$$

where $v \in [0, 1]$ denotes the FLIP blending parameter, and

$$\Delta \mathbf{u}_{n+1,\tau}^s(\mathbf{x}) = \mathbf{u}_{n+1,\tau}^s(\mathbf{x}) - \mathbf{u}_{n,\tau}^s(\mathbf{x}),$$

is interpolated to particles using Eq. 7. We have used $v = 0.95$ for all of our examples.

6.7 Advection

Trajectories for all advection processes in both phases are computed, for simplicity, using forward Euler and trilinear interpolation. In spatially adaptive domains we follow the method of Ando and Batty [2020].

Ambient grid density and velocity are backtraced with semi-Lagrangian advection. If a backtrace position ends up outside of the simulated domain Ω , we sample the user-provided functions $\rho_{\partial\Omega}^a, \mathbf{u}_{\partial\Omega}^a$ (App. A).

Submerged particles are forward-advected according to submerged velocities on the grid. After advection, any particle $q \in \mathcal{P}$ inside a boundary $\Phi^{\text{solid}}(\mathbf{x}_q) < 0$ is moved to the closest point outside of the boundary, $\mathbf{x}_q - \Phi^{\text{solid}} \nabla \Phi^{\text{solid}}$.

7 Implementation

For visualization we primarily render the submerged phase geometry directly without any post-processing. Depending on the scenario, this consists of rendering either the particles or the submerged fraction field. Two exceptions are the geyser (Fig. 3) and the barrel wave (Fig. 11), where we separate particles based on the dispersed threshold ϕ^{UR} . Particles above the threshold are surfaced using the method of Lesser et al. [2022], while particles below the threshold are rasterized into a volumetric density field and rendered as smoke. Critically however, we never add any additional spray or bubble particles or material fields beyond those that directly participated in the main simulations.

Fluxed boundaries. We leverage the fluxed animated boundary method [Stomakhin and Selle 2017] to define velocity boundary conditions within our solver. While we could, in principle, define independent boundary conditions for each phase, we instead enforce a shared boundary velocity for both the submerged and ambient phases.

Particle initialization. Since our formulation operates on a mixture, we prescribe the composition of the emitted flow in terms of a submerged emission fraction. This ratio determines the amount of submerged material introduced at the start of the simulation or at a fluxed boundary, and is used to generate particles for the submerged phase whenever the emission fraction is non-zero.

7.1 Particle splitting strategies

To leverage the fact that our method supports varying particle sizes, we consider two strategies for dynamically determining the particle size during runtime. The goal is to increase resolution in regions

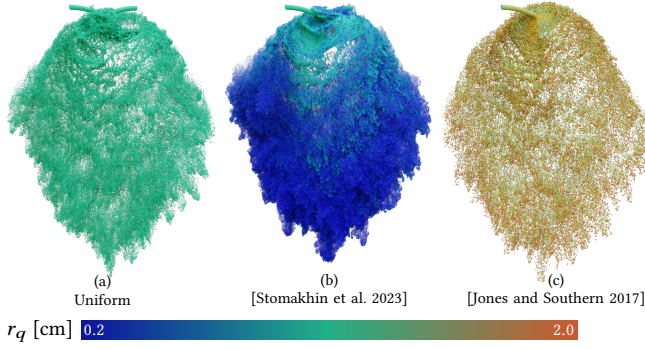


Fig. 12. *Splitting heuristic*. Two liquid streams collide, spraying water over a large region. We compare the effect of different splitting heuristics (§7.1) on the resulting dynamics. Particles are colored by size, using the same colormap across figures, and scaled by a factor of $3\times$ for improved visibility. (a) Baseline simulation without splitting. (b) Volumetric splitting heuristic, resulting in a $50\times$ increase in particle count (Table. 3). (c) Particle-based heuristic with both splitting and merging. Visually, we have preferred (b). ©Wētā FX

where the flow becomes dispersed, while maintaining a coarser representation elsewhere.

Volume-based splitting. We implement a particle splitting algorithm loosely based on that of Stomakhin et al. [2023, §4.2.1]. A particle q is marked for splitting if it is classified as dispersed and experiences sufficiently large relative phase motion,

$$\phi^s(\mathbf{x}_q) < \phi^{UR} \quad \wedge \quad \|\mathbf{u}^s(\mathbf{x}_q) - \mathbf{u}^a(\mathbf{x}_q)\| > \psi^u.$$

Splitting is suppressed for recently created particles (age < 0.05 s), or if the splitting would produce particles below a minimum volume threshold (set to 0.05 times the initial particle volume).

Marked particles are split into two daughter particles. The volume is divided by sampling a uniform random variable, and particle positions are perturbed slightly to avoid overlap. The resulting particles inherit velocities by sampling the submerged velocity field, setting $\mathbf{u}_q^s \leftarrow \mathbf{u}^s(\mathbf{x}_q)$.

Particle-based splitting and merging. As an alternative, we implement the particle-based splitting and merging strategy of Jones and Southern [2017]. Particle interactions are detected through intersection of particle trajectories and classified as coalescing, separating, or fragmenting based on an experimentally validated collision model. This formulation operates directly on particle interactions rather than continuous mixture equations.

We compare the effects of these strategies in Fig. 12. Visually, we have preferred the volumetric splitting, and use it for our larger waterfall (Fig. 1, Fig. 17) and barrel wave (Fig. 11) examples.

7.2 Spatial grid adaptivity

By decoupling particles size from the grid resolution, our method naturally supports spatially adaptive grids, which we have adopted in this work. To this end, we introduce buckets (see App. A) that vary in sizes that are power-of-two multiples of a base scale. To maintain a well-graded hierarchy, size changes between adjacent

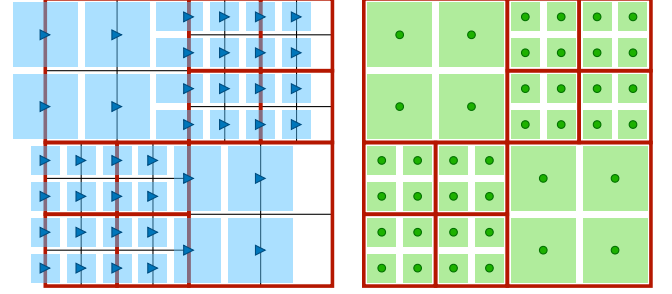


Fig. 13. *Spatially adaptive interpolants*. Support of N_t^0 (right) and a single component of N_t^0 (left) in a spatially adaptive setting. Red boxes represent sparse buckets of varying sizes (App. A), each subdivided into 2×2 voxels. The non-zero region of the interpolant is visualized by the transparent colors (blue faces; green cells) associated with the contained cell (circle) or face (triangle). Near resolution jumps, the face interpolant N_t^0 must be stretched/compressed to ensure partition of unity; no such correction is necessary for cells. ©Wētā FX

buckets are restricted to be at most a factor of two. All volume-based operations are carried out similar to Ando and Batty [2020].

To enable particle-grid transfers on adaptive domains, we need to make sure our grid interpolants maintain the partition of unity property near resolution jumps, similar to Gao et al. [2017]. We focus on the zeroth-order grid interpolant N^0 which, combined with the particle box kernel Λ^0 , yields multilinear weights sufficient for our grid-based pressure projection; other variants are left for future work. For cell-centered quantities, no special treatment is required, as the grid kernels coincide with cell indicator functions. For face-centered quantities, however, kernels are modified near resolution changes—becoming rectangular where needed—to avoid gaps or overlaps; see Fig. 13.

8 Results and discussion

All simulations were performed on a cluster of machines with varying characteristics. Most scenarios were run on nodes with 32 physical CPU cores and 256 GB of RAM. Detailed simulation parameters and runtimes for each example are reported in Tab. 3.

We evaluate our method on a range of scenarios spanning stratified, dispersed, and multi-scale flow features (refer to our supplemental video for the resulting animations). In contrast to prior approaches that typically target one regime, our unified formulation allows these diverse behaviours to emerge naturally from the local flow conditions.

Unless otherwise stated, simulations use a baseline configuration of 8 particles per voxel (PPV), an emission fraction of 1 resulting in particle volume $V_q = h^3/\text{PPV}$, and a uniform grid with voxel size h . The barrel wave (Fig. 11) and waterfall (Fig. 1) examples employ the volume-based splitting strategy outlined in §7.1. The geyser inlet (Fig. 3) uses a reduced emission fraction of 0.1 to mimic vapour in the ejected mixture, corresponding to smaller particle sizes (i.e., $V_q = 0.1h^3/\text{PPV}$). Similarly, the dispersed bubble tank (Fig. 14) uses a reduced emission fraction of 0.02, and provides a comparison of using equal-sized particles versus the stochastic initialization of

Wretborn et al. [2022, §5.2]. The waterbombing example (Fig. 4) uses an adaptive grid with four levels; emission is performed with 64 PPV.

Qualitative results. Two-fluid models have traditionally been applied to dispersed regimes (e.g., [Nielsen and Østerby 2013]). We demonstrate this capability within our formulation by recreating the bubble column from Kim et al. [2010, Fig. 5] in Fig. 14. Our multi-scale representation further extends to stratified flows, as seen in the bubble barrel example (Fig. 5; cf. [Wretborn et al. 2025, Fig. 15]).

The coastal cliffs scene (Fig. 2) represents a large-scale scenario commonly simulated using single-phase sharp-interface methods. Although the particles are of a similar size to those in standard methods (e.g., FLIP), our sub-grid model interprets them as aggregates of smaller droplets, amplified by the small dispersed radius scale $\psi^r = 0.01$. As observed by Braun et al. [2025], simulating the air phase in such scenarios is essential for conveying a sense of scale.

Several examples demonstrate continuous transitions between flow regimes. In the geyser (Fig. 3), the liquid jet rapidly breaks up, transitioning from a collimated column into a dispersed spray. In the waterbombing example (Fig. 4), the leading edge develops characteristic liquid-gas instabilities, leaving behind a diffuse wake. The barrel wave (Fig. 11) initially forms a sharp interface, which fragments upon impact as particles are split (§7.1), producing both stratified and dispersed structures. Similarly, the waterfall (Fig. 1) exhibits a transition from a smooth sheet near the outlet to fine spray as it falls.

The key to these results is in our methodology, and particularly our formulation of two-fluid closure terms (§6.3). It is this framework that allows for different simulation regions to behave like a dispersed spray or a stratified liquid, or a continuous blend between the two. For example, even though the waterfall (Fig. 1) uses a fairly modest voxel size of 25 cm, it can still capture both the effect of fine mist as well as a smooth sheet of water simultaneously, without a change in discrete representation.

Comparison to existing methods. In Figure 16, we compare our method to several common particle-based approaches to analyze

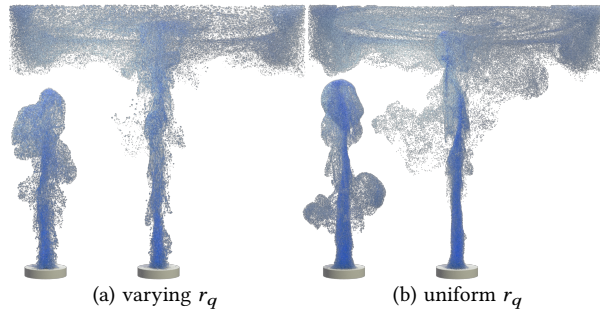


Fig. 14. *Bubble column.* We release bubbles in a $30 \times 30 \times 30$ cm³ box for (a) varying and (b) uniform particle size distributions, keeping other simulation parameters fixed. Snapshots at equal times are shown for the two simulations. Bubble sizes influence dynamics via our dispersed closure modeling (§6.3). The simulation remains stable at a 1:1000 density ratio, achieving similar qualitative effects as Kim et al. [2010]. ©Wētā FX

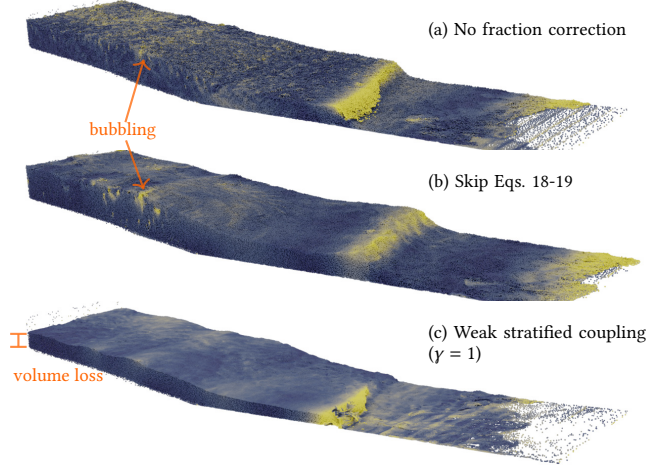


Fig. 15. *Method ablation study.* We remove key components of our solver to study their effect: (a) omitting the fraction correction procedure (§6.2) leads to severe, spurious bubbling throughout the domain; (b) omitting the fraction correction extrapolation procedure (Eqs. 18-19) causes bubbling near boundaries; (c) if the relaxation time γ in the stratified closure term (Eq. 22) is chosen “too small”, the weak phase coupling causes loss of submerged liquid volume. Particles are colored by speed. ©Wētā FX

the ability of each to capture stratified flows. All solvers use FLIP [Brackbill and Ruppel 1986] for particle velocity updates with the same blending parameter ($\nu = 0.95$).

We begin with two homogenized formulations, where a single velocity field governs both phases. For simplicity, these versions use a dense domain enclosed by the dam walls; all other methods use a sparse domain representation (App. A). Practically, these approaches were implemented by computing a density field for the mixture $\rho = \phi^s \rho^s + \phi^a \rho^a$ and treating the resulting pressure system (§6.4) as single-phase. First, in the two-phase marker variant (Fig. 16a), particles populate the entire domain with densities $\rho_q \in \{1, 1000\}$, resulting in $\phi^a = 0$ everywhere. Second, in the submerged marker variant (Fig. 16b), particles only represent the submerged phase (i.e., $\rho_q = 1000$) and we use the same rasterization procedure from §6.1. In both cases, the single-velocity assumption leads to excessive numerical mixing, and dispersed regions are slow to unmix into cleanly separated phases.

Sharp-interface methods (Fig. 16c) avoid this issue by explicitly reconstructing a free surface using level sets, enabling stable stratified flows and allowing the two phases to cleanly unmix. However, they cannot capture dispersed regimes.

Diffuse interface variants (Fig. 16d) smooth the reconstructed interface, but still rely on a single velocity field in the mixture region. We implement a particle phase-indicator variant (see terminology in §2) following Kim et al. [2010, §3.2], who convert a sharp level set to a single-phase variable-density fluid; we mimic this by reconstructing the level set from particles. This conversion preserves a geometrically defined interface, in contrast to purely homogenized formulations, and thereby improves the ability of the two fluids to unmix (cf. Figs. 16a-16b). This helps explain the effectiveness

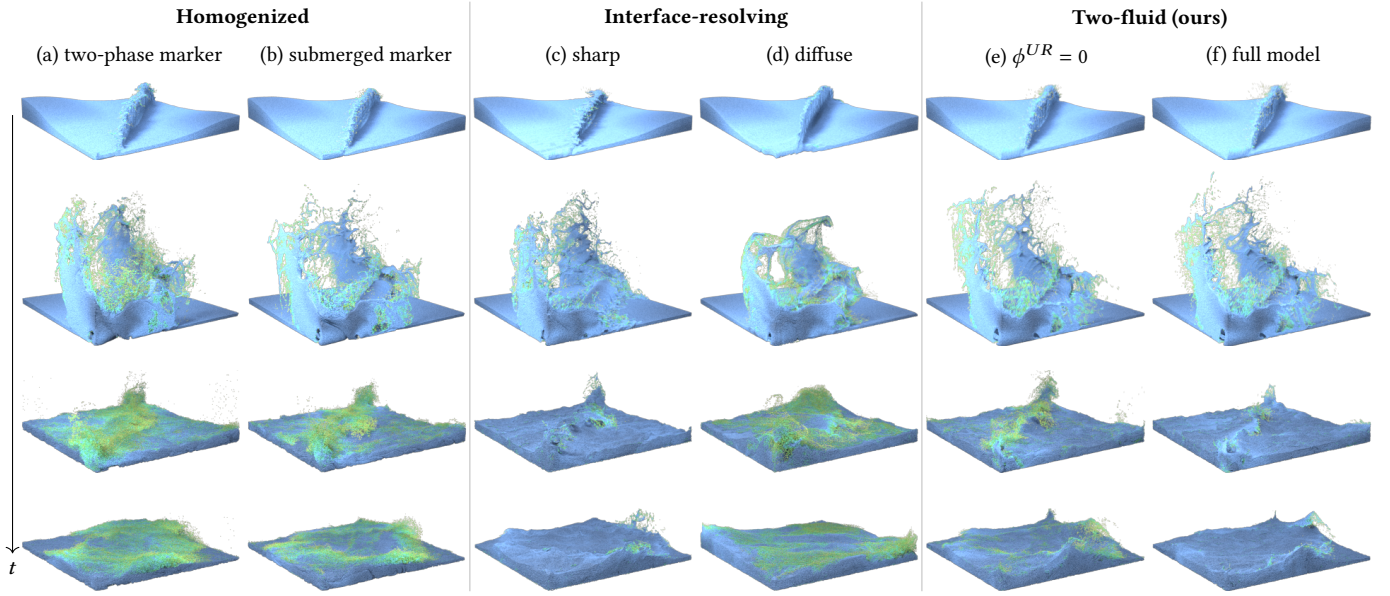


Fig. 16. *Solver comparison.* We compare how well six particle-based two-phase solvers perform a double dam-break, focusing on their ability to recover separated phases after strong mixing. Particles are colored based on their density (hue) and speed (brightness): blue corresponds to approximately 8 particles per voxel, while green indicates lower densities. Our two-fluid formulation (Figs. 16e, 16f) enables a similar phase separation to sharp-interface methods (see Fig. 16c), while homogenized and diffuse methods exhibit reduced ability to unmix (see Figs. 16a, 16b, 16d). ©Wētā FX

of related approaches such as Braun et al. [2025]: while they do not explicitly use a level set for reconstruction, their sharpening procedure [Braun et al. 2025, §3.3] fills a similar role. However, in our experience, these diffuse variants still exhibit more numerical mixing than sharp, interface-resolving methods. Similar artificial mixing artifacts, due to errors near a diffuse interface, have been reported previously [Ando et al. 2015, Fig. 7].

Introducing separate velocity fields for each phase fundamentally changes this behaviour. In the two-field variant without a sub-grid closure model (Fig. 16e; i.e., our method with $\phi^{UR} = 0$), unmixing emerges naturally from the model, even when strong closure forces enforce near no-slip behaviour between phases. This indicates that phase separation is no longer a geometric or numerical artifact, but a consequence of the governing equations. Finally, when the sub-grid model is enabled (Fig. 16f), our full method captures both dispersed and stratified dynamics, while allowing the two phases to unmix.

Thus, our method demonstrates strong overall qualitative performance. Let us next make a closer inspection of several further aspects of our method.

Correction of the submerged fraction field. The fraction correction procedure (§6.2) is heuristic and not derived from first principles. However, similar interface regularization is typical in interface-averaging methods in the literature. In particular, several existing Eulerian two-fluid approaches introduce artificial sharpening terms to avoid diffusion of stratified features (e.g., see the list presented by De Santis et al. [2021, Table 1]).

Our formulation performs two corrections: removing nonphysical numerical oscillations in stratified regions where $\bar{\phi}^s \approx 1$, and extrapolating fractions inside boundaries. As demonstrated in Fig. 15, omitting either step leads to spurious small-scale fluctuations that manifest as artificial bubbling.

We note that the correction procedure partially relaxes the interpretation of particles as strict carriers of fluid volume. However, this deviation is confined to stratified regions and near boundaries, where the correction is applied. In dispersed regimes, the correspondence remains exact, and particles represent submerged fluid volume consistently.

Quality of stratified features. A natural limitation of the proposed approach is that, as an interface-averaging method, it does not achieve the same accuracy near resolved interfaces as sharp-interface techniques using, for example, the ghost fluid method [Fedkiw et al. 1999].

However, the quality of the reconstructed interface can be improved in several ways. Increasing the order of the grid interpolant or reconstructing the fraction field on a higher-resolution grid both yield smoother interfaces, as shown in Fig. 10. A potential concern is that wider interpolants may introduce artifacts similar to those observed in the material point method, where contact locations are biased by the support of the interpolation kernel [Wretborn et al. 2017, §6]. In our experiments, we observed no such effects (Fig. 18).

We do observe one numerical issue: a mild cell-crossing instability can occur near non-axis-aligned boundaries for very thin interfaces ($< h$); see the top-row of Fig. 15. This effect is exacerbated when combining the Λ^0 particle basis with a zeroth-order interpolant

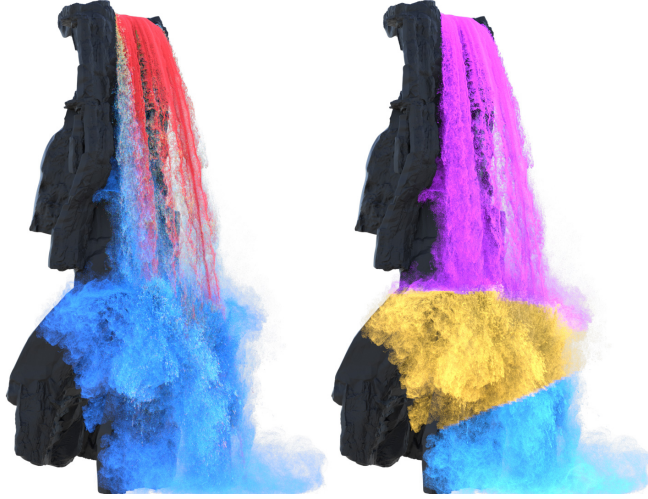


Fig. 17. *Waterfall breakdown.* A diagnostic view of the simulation in Fig. 1, illustrating particle distribution and domain decomposition. The simulation is distributed across three machines using the partitioner of Wretborn et al. [2026], and uses the volumetric splitting strategy (§7.1). Left: particles colored by size (red: large; blue: small). Right: particles colored by owning rank. ©Wētā FX

N^0 . The effect is mitigated by using smoother interpolants and higher-resolution fraction fields.

Effect of model parameters. Figure 9 illustrates the influence of the key model parameters. In this experiment, we keep the grid resolution fixed while varying the mixture composition and effective droplet size by adjusting the particle volume and the dispersed radius scale ψ^r .

Reducing the particle size decreases the local submerged fraction, shifting the flow toward a more dispersed regime. At the same time, the effective droplet radius, given by $\psi^r r_q$, controls the strength of the sub-grid closure terms (Eqs. 28-30; see also Figs. 8 and 19a). Smaller effective radii produce relatively stronger drag (since the drag force density is inversely proportional to the radius), while larger values allow for greater inter-phase slip. Together, these parameters determine the transition between stratified and dispersed behaviour.

The nonlinear drag coefficient ψ^{NL} primarily affects fast-moving structures. Increasing this coefficient increases resistance and promotes fluid break-up, while lower values result in more ballistic motion. We keep this coefficient fixed at $\psi^{NL} = 1$ in all examples.

The virtual mass term (Eq. 32) has little effect for heavy submerged material in light ambient fluids, but is essential for stable simulation of light submerged materials. For example, if the virtual mass force is disabled (i.e., $\psi^{VM} = 0$), simulations of dispersed bubbles (e.g., Fig. 14) become unstable immediately. Since the virtual mass term is relatively cheap to compute, it is enabled in all experiments.

Adaptive particle sizes. A key feature of our formulation is that particle size is decoupled from the underlying grid resolution. As demonstrated in the waterfall example (Fig. 1), this allows us to represent a wide range of scales within a single simulation by varying

particle size (as visualized in Fig. 17) independently of the Eulerian discretization.

However, particle size also influences the dynamics in dispersed (but not stratified) regions through the sub-grid closure model. As a result, particle size is a modeling parameter that must be chosen carefully. For physically accurate simulations, this necessitates atomization criteria derived from fluid properties.

In this work, we use a heuristic splitting strategy inspired by Stomakhin et al. [2023]. While effective in practice, these approaches lack a firm physical basis and are subject to parameter tuning based on the scenario. We have also explored the breakup model of Jones and Southern [2017], seen in Fig. 12. While grounded in physics, by being particle-based it deviates from the otherwise Eulerian nature of our formulation, and it is unclear how to adapt it when particles represent aggregates of fluid rather than individual droplets.

Similarly, particle merging remains an open problem. Existing probabilistic merging strategies [Braun et al. 2025; Stomakhin et al. 2023] introduce mass loss, which is undesirable in our setting. Developing physically grounded adaptive particle strategies is an important avenue for future work.

Performance. Our method achieves performance comparable to a sharp-interface two-phase FLIP solver (e.g., Boyd and Bridson [2012]). Additional costs stem primarily from rasterizing more quantities and, when using a larger particle support, increased stencil sizes. Closure terms are inexpensive to evaluate, except for the use of the dispersed regime indicator (Eq. 27) which requires one extra rasterization pass. The closure-aware, variable-coefficient pressure projection (§6.4) does not, in our experience, introduce a noticeable performance penalty compared to a constant-coefficient sharp-interface version. Timings are provided in Table 3.

9 Future work

Exotic sub-grid models. Our proposed two-fluid formulation naturally supports closure models expressed in terms of relative phase velocity. Extending the framework to incorporate more sophisticated sub-grid terms—such as lift forces or turbulent dispersion—remains an interesting direction for future work. It may also be possible to parameterize these interactions to provide practitioners with intuitive control over two-phase behavior, enabling targeted manipulation of interfacial dynamics (e.g., breakup and spray formation) that is otherwise difficult to achieve.

Adaptivity heuristics. The adaptivity levels in the waterbombing example (Fig. 4) are currently prescribed. While particle size is decoupled from grid resolution—eliminating the need for reseeded across levels—it directly affects dynamics in dispersed regimes through the drag closure terms (Eqs. 29-30). This interaction introduces a nontrivial trade-off, as different particle sizes yield different effective behavior. An interesting direction for future work is the design of adaptive policies that jointly select particle size and grid resolution based on local flow conditions.

Submerged volumetric mist. When using splitting techniques (§7.1), the number of small particles can proliferate, increasing computational cost while contributing little to the overall dynamics. A promising direction is to replace such particles with a coarse-grained

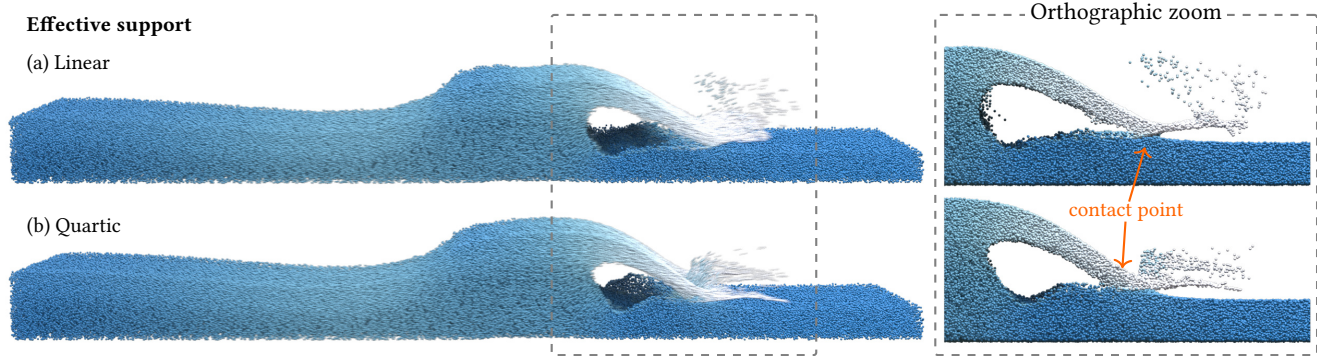


Fig. 18. *Support comparison.* We compare the effect of particle support on submerged phase self-interactions. All simulation parameters are identical except for the rasterization configuration (§5.2): (a) N^0 with $\tilde{\Lambda}^0$; (b) N^3 with Λ^0 . Right: although the effective support in (b) spans multiple grid cells, as in (a), no visible separation of the contact point is observed. This space-separation artifact, common in material point methods (§8), is not observed in our method. ©Wētā FX

volumetric representation. In the two-fluid framework of Frederix et al. [2021], this effect is achieved by introducing multiple numerical fields per physical phase with morphological transition heuristics. A similar approach could be naturally integrated into our system to, for example, represent submerged mist using a volumetric field.

Fraction correction alternatives. Our current correction procedure (§6.2) operates locally at each time step. An alternative approach is to track the submerged phase temporally—either by advecting the fractions from the previous frame, or by attaching information on the particles as done by Goldade et al. [2020]—and use this information to correct the submerged fraction. This could be combined with adaptive reconstruction strategies such as that of Ferstl et al. [2016].

10 Conclusion

We have presented a unified, multi-scale method for simulating both stratified and dispersed two-phase flow regimes within a single framework. Our approach is based on a two-fluid mixture formulation, in which phases exchange forces through sub-grid closure models that are solved for in a strongly coupled manner using a pressure projection step. The primary (submerged) phase is represented using particles, whose properties are rasterized to the grid via GIMP, enabling particle size to be decoupled from the grid resolution.

This unified treatment offers a fundamentally different alternative to traditional interface-resolving approaches commonly used in computer graphics. As we demonstrated across a range of scenarios, the method is particularly well-suited to large-scale scenarios in which stratified and dispersed regimes coexist, yet avoids the need for multiple distinct solvers or passes. We therefore believe it provides a practical and often more robust choice for many graphics applications, where such multi-scale behavior is prevalent.

11 Acknowledgements

The authors extend their gratitude to the Wētā FX leadership and Simulation department for supporting this work, as well as to the anonymous reviewers for useful their feedback. This research was supported in part by grants from NSERC (RGPIN-2021-02524 and

RGPIN-2026-04447) and CFI-JELF (Grant 40132). We thank Rob Tuytel and Rico Cilliers for the coastal cliffs model [Poly Haven 2024], Airbus5717 for the airplane model [Sketchfab 2021], and Kevin Blom for his help setting up the coastal cliffs scene.

References

- Ryoichi Ando and Christopher Batty. 2020. A practical octree liquid simulator with adaptive surface resolution. *ACM Trans. Graph.* 39, 4 (Aug. 2020), 32:1–32:17.
- Ryoichi Ando, Nils Thuerey, and Chris Wojtan. 2015. A stream function solver for liquid simulations. *ACM Trans. Graph.* 34, 4 (July 2015), 1–9.
- Scott G Bardenhagen, Edward M Kober, and Others. 2004. The generalized interpolation material point method. *Computer Modeling in Engineering and Sciences* 5, 6 (2004), 477–496.
- Landon Boyd and Robert Bridson. 2012. MultiFLIP for energetic two-phase fluid simulation. *ACM Trans. Graph.* 31, 2 (April 2012), 1–12.
- J U Brackbill and H M Ruppel. 1986. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *J. Comput. Phys.* 65, 2 (Aug. 1986), 314–343.
- Bernhard Braun, Jan Bender, and Nils Thuerey. 2025. Adaptive phase-field-FLIP for very large scale two-phase fluid simulation. *ACM Transactions on Graphics* 44, 4 (2025).
- Robert Bridson. 2015. *Fluid Simulation for Computer Graphics*. CRC Press.
- A De Santis, M Colombo, B C Hanson, and M Fairweather. 2021. A generalized multi-phase modelling approach for multiscale flows. *J. Comput. Phys.* 436, 110321 (July 2021), 110321.
- Ronald P Fedkiw, Tariq Aslam, Barry Merriman, and Stanley Osher. 1999. A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method). *J. Comput. Phys.* 152, 2 (July 1999), 457–492.
- Yun (Raymond) Fei, Christopher Batty, Eitan Grinspun, and Changxi Zheng. 2018. A multi-scale model for simulating liquid-fabric interactions. *ACM Trans. Graph.* 37, 4 (July 2018), 1–16.
- Florian Ferstl, Ryoichi Ando, Chris Wojtan, Rüdiger Westermann, and Nils Thuerey. 2016. Narrow band FLIP for liquid simulations. *Comput. Graph. Forum* 35, 2 (May 2016), 225–232.
- Sean Flynn, Alexey Stomakhin, Joel Wretborn, Noh-Hoon Lee, Job Guidos, Nicholas Illingworth, James Robinson, and Ryan Bowden. 2026. Advancing large-scale water simulation in avatar: Fire and ash. In *ACM SIGGRAPH 2026 Talks*. New York, NY, USA, 1–3.
- E M A Frederix, D Dovizio, A Mathur, and E M J Komen. 2021. All-regime two-phase flow modeling using a novel four-field large interface simulation approach. *Int. J. Multiph. Flow* 145, 103822 (Dec. 2021), 103822.
- Ming Gao, Andre Pradhana Tampubolon, Chenfanfu Jiang, and Eftychios Sfakias. 2017. An adaptive generalized interpolation material point method for simulating elastoplastic materials. *ACM Trans. Graph.* 36, 6 (Nov. 2017), 1–12.
- Ryan Goldade, Mridul Aanjaneya, and Christopher Batty. 2020. Constraint bubbles and affine regions: reduced fluid models for efficient immersed bubbles and flexible spatial coarsening. *ACM Trans. Graph.* 39, 4 (July 2020), 43:1–43:15.
- Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The affine particle-in-cell method. *ACM Trans. Graph.* 34, 4 (July 2015), 1–10.

- Richard Jones and Richard Southern. 2017. Physically-based droplet interaction. In *Eurographics*. ACM, New York, NY, USA.
- Petr Karnakov, Sergey Litvinov, and Petros Koumoutsakos. 2022. Computing foaming flows across scales. From breaking waves to microfluidics. *Sci Adv* 8, 5 (Feb. 2022).
- Doyub Kim, Oh-Young Song, and Hyeon-Seok Ko. 2010. A practical simulation of dispersed bubble flow (*SIGGRAPH '10, Article 70*). New York, NY, USA, 1–5.
- Steve Lesser, Alexey Stomakhin, Gilles Daviet, Joel Wretborn, John Edholm, Noh-Hoon Lee, Eston Schweickart, Xiao Zhai, Sean Flynn, and Andrew Moffat. 2022. Loki: a unified multiphysics simulation framework for production. *ACM Trans. Graph.* 41, 4 (July 2022), 1–20.
- Wei Li, Kui Wu, and Mathieu Desbrun. 2024. Kinetic simulation of turbulent multifluid flows. *ACM Trans. Graph.* 43, 4 (July 2024), 1–17.
- Frank Losasso, Jerry Talton, Nipun Kwatra, and Ronald Fedkiw. 2008. Two-way coupled SPH and particle level set fluid simulation. *IEEE TVCG* 14, 4 (2008), 797–804.
- Richard Meller, Fabian Schlegel, and Dirk Lucas. 2021. Basic verification of a numerical framework applied to a morphology adaptive multifield two-fluid model considering bubble motions. *Int. J. Numer. Methods Fluids* 93, 3 (March 2021), 748–773.
- Richard Meller, Matej Tekavcic, Benjamin Krull, and Fabian Schlegel. 2023. Momentum exchange modeling for coarsely resolved interfaces in a multifield two-fluid model. *Int. J. Numer. Methods Fluids* 95, 9 (Sept. 2023), 1521–1545.
- Efstathios E Michaelides, Martin Sommerfeld, and Berend van Wachem. 2022. *Multiphase flows with droplets and particles, third edition* (3 ed.). CRC Press, Boca Raton.
- Ken Museth. 2013. VDB: High-resolution sparse volumes with dynamic topology. *ACM Trans. Graph.* 32, 3 (June 2013), 1–22.
- Michael B Nielsen and Ole Østerby. 2013. A two-continua approach to Eulerian simulation of water spray. *ACM Trans. Graph.* 32, 4 (July 2013), 1–10.
- Stanley Osher and Ronald Fedkiw (Eds.). 2002. *Level set methods and dynamic implicit surfaces*. Springer New York, NY.
- Saket Patkar, Mridul Aanjaneya, Dmitriy Karpman, and Ronald Fedkiw. 2013. A hybrid Lagrangian-Eulerian formulation for bubble generation and dynamics (*SCA '13*). New York, NY, USA, 105–114.
- Poly Haven. 2024. Coastal Cliff 04. https://polyhaven.com/a/coastal_cliff_04 CC0.
- Stéphane Popinet. 2009. An accurate adaptive solver for surface-tension-driven interfacial flows. *J. Comput. Phys.* 228, 16 (Sept. 2009), 5838–5866.
- Bo Ren, Chenfeng Li, Xiao Yan, Ming C Lin, Javier Bonet, and Shi-Min Hu. 2014. Multiple-fluid SPH simulation using a mixture model. *ACM Trans. Graph.* 33, 5 (Sept. 2014), 1–11.
- Sketchfab. 2021. Airbus A320-200. <https://sketchfab.com/3d-models/airbus-a320-200-7b8910d3cf824e6399c4ca75a31130ed> Sketchfab, CC BY-NC.
- Jos Stam. 1999. Stable fluids. In *SIGGRAPH '99*. ACM Press, New York, New York, USA.
- Alexey Stomakhin, Steve Lesser, Joel Wretborn, Sean Flynn, Johnathan Nixon, Nicholas Illingworth, Adrien Rollet, Kevin Blom, and Douglas McHale. 2023. Pahi: A Unified Water Pipeline and Toolset (*DigiPro '23, Article 11*). New York, NY, USA, 1–13.
- Alexey Stomakhin and Andrew Selle. 2017. Fluxed animated boundary method. *ACM Trans. Graph.* 36, 4 (July 2017), 1–8.
- Alexey Stomakhin, Joel Wretborn, Kevin Blom, and Gilles Daviet. 2020. Underwater bubbles and coupling. In *ACM SIGGRAPH 2020 Talks*. ACM, New York, NY, USA.
- Joel Wretborn, Rickard Armiento, and Ken Museth. 2017. Animation of crack propagation by means of an extended multi-body solver for the material point method. *Comput. Graph.* 69 (Dec. 2017), 131–139.
- Joel Wretborn, Sean Flynn, and Alexey Stomakhin. 2022. Guided bubbles and wet foam for realistic whitewater simulation. *ACM Trans. Graph.* 41, 4 (July 2022), 1–16.
- Joel Wretborn, Marcus Schoo, Noh-Hoon Lee, Christopher Batty, and Alexey Stomakhin. 2026. A practical partitioner for distributed simulations on sparse dynamic domains using optimal transport. *ACM Trans. Graph.* 3787521 (Jan. 2026).
- Joel Wretborn, Alexey Stomakhin, and Christopher Batty. 2025. A unified multi-scale method for simulating immersed bubbles. *Comput. Graph. Forum* (April 2025).
- Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Trans. Graph.* 24, 3 (July 2005), 965–972.
- L Štrubelj and I Tiselj. 2011. Two-fluid model with interface sharpening. *Int. J. Numer. Methods Eng.* 85, 5 (Feb. 2011), 575–590.

A Sparse dynamic simulation domain

We employ a block-structured sparse grid, similar in spirit to OpenVDB [Museth 2013], and we will borrow the mathematical notation from Wretborn et al. [2026, §2] to formalize how the discrete domain is defined. Let $\mathcal{B} : \mathbb{R}^3 \rightarrow B$ be a *bucketization* function, which maps every point in space to a unique bucket $b \in B$ (using OpenVDB terminology, a b would be called a “leaf node”). We define the set of buckets covering all particles at time n ,

$$D_n^s = \{\cup_{q \in \mathcal{P}} \mathcal{B}(\mathbf{x}_{n,q})\}. \quad (46)$$

We call D_n^s the *submerged simulation domain subset*. In practice, each bucket will be further subdivided into voxels (commonly by $4^3 - 8^3$ per bucket).

The *ambient* simulation domain subset, D_n^a , is defined to guarantee that it fully surrounds the submerged phase. Let $\mathcal{T} : 2^B \rightarrow 2^B$ be the function that dilates a set of buckets by a single layer [Wretborn et al. 2026, Eq. 2], and let

$$D_n^a = \mathcal{T}_k(D_n^s) \equiv \underbrace{(\mathcal{T} \circ \dots \circ \mathcal{T})}_{k \text{ dilations}}(D_n^s). \quad (47)$$

Equation 47 ensures that the ambient fluid is only simulated in a localized region around the submerged phase, and $k \in \mathbb{N}$ makes it possible to expand the simulation domain by performing the dilation multiple times. We denote region of space covered by all buckets in the ambient simulation domain subset with $\Omega \subset \mathbb{R}^3$; that is, Ω represents the spatial extent of the simulation domain.

Ambient border boundary conditions. The ambient simulation domain D_n^a represents a finite cutout of an effectively infinite ambient fluid, and the user must define ambient density $\rho_{\partial\Omega}^a(t, \mathbf{x})$ and ambient velocity $\mathbf{u}_{\partial\Omega}^a(t, \mathbf{x})$ anywhere the simulation may end up.

Evolving the domain topology. The simulation domains D_n^s and D_n^a will change in time. We initialize ambient phase quantities in newly created buckets $D_n^a \setminus D_{n-1}^a$ using the user-provided functions $\rho_{\partial\Omega}^a$ and $\mathbf{u}_{\partial\Omega}^a$.

B Pressure projection with closure terms

We show here a derivation of the block-diagonalized equations of motion, Eq. 36. Our starting point is the discrete equations of motion (Eq. 35):

$$\begin{pmatrix} [m_{\mathcal{T}}^s + \Delta t D_{\mathcal{T}}] & -\Delta t [D_{\mathcal{T}}] & \Delta t [\phi_{\mathcal{T}}^s] \mathbf{G} \\ -\Delta t [D_{\mathcal{T}}] & [m_{\mathcal{T}}^a + \Delta t D_{\mathcal{T}}] & \Delta t [\phi_{\mathcal{T}}^a] \mathbf{G} \\ \mathbf{G}^T [\phi_{\mathcal{T}}^s] & \mathbf{G}^T [\phi_{\mathcal{T}}^a] & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\mathcal{T}}^s \\ \mathbf{u}_{\mathcal{T}}^a \\ p_C \end{pmatrix} = \begin{pmatrix} [m_{\mathcal{T}}^s] \bar{\mathbf{u}}_{\mathcal{T}}^s \\ [m_{\mathcal{T}}^a] \bar{\mathbf{u}}_{\mathcal{T}}^a \\ 0 \end{pmatrix}.$$

We sum rows one and two,

$$\begin{pmatrix} [m_{\mathcal{T}}^s] & [m_{\mathcal{T}}^a] & \Delta t ([\phi_{\mathcal{T}}^s] + [\phi_{\mathcal{T}}^a]) \mathbf{G} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\mathcal{T}}^s \\ \mathbf{u}_{\mathcal{T}}^a \\ p_C \end{pmatrix} = \begin{pmatrix} [m_{\mathcal{T}}^s] \bar{\mathbf{u}}_{\mathcal{T}}^s \\ [m_{\mathcal{T}}^a] \bar{\mathbf{u}}_{\mathcal{T}}^a \\ 0 \end{pmatrix}.$$

$$\begin{pmatrix} [m_{\mathcal{T}}^s] & [m_{\mathcal{T}}^a] & \Delta t \mathbf{G} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\mathcal{T}}^s \\ \mathbf{u}_{\mathcal{T}}^a \\ p_C \end{pmatrix} = [m_{\mathcal{T}}^s] \bar{\mathbf{u}}_{\mathcal{T}}^s + [m_{\mathcal{T}}^a] \bar{\mathbf{u}}_{\mathcal{T}}^a, \quad (48)$$

where we used the fact that $[\phi_{\mathcal{T}}^s] + [\phi_{\mathcal{T}}^a] = \mathbb{I}_{\mathcal{T}}$ (Eq. 4). Equation 48 can be used in turn to eliminate the off-diagonal velocity terms in Eq. 35.

We will demonstrate this procedure for the first row in Eq. 35. Multiply Eq. 48 with $\Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1}$ and add it to the first row in Eq. 35. To save horizontal space, we will compute one component at a time.

- Component multiplying $\mathbf{u}_{\mathcal{T}}^s$:

$$\begin{aligned} & \Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} [m_{\mathcal{T}}^s] + [m_{\mathcal{T}}^s + \Delta t D_{\mathcal{T}}] = \\ & = \Delta t [D_{\mathcal{T}}] \left(\mathbb{I}_{\mathcal{T}} + [m_{\mathcal{T}}^a]^{-1} [m_{\mathcal{T}}^s] \right) + [m_{\mathcal{T}}^s] \equiv \mathbf{M}^s. \end{aligned} \quad (49)$$

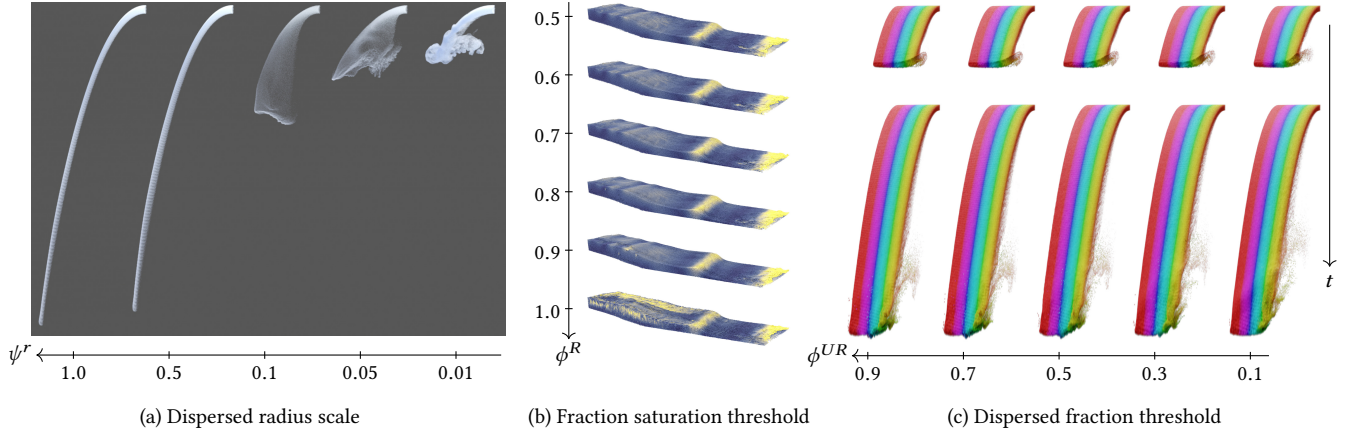


Fig. 19. *Parameter studies.* We vary selected model parameters: (a) the dispersed radius scale ψ^r strongly affects dynamics in the dispersed regime; (b) large saturation thresholds ϕ^R can produce bubbling artifacts; (c) the dispersed fraction threshold ϕ^{UR} controls the classification of regions as dispersed (§6.3). In our solver, ψ^r is user-controlled, while $\phi^R = 0.5$ and $\phi^{UR} = 0.3$ are held fixed by default. Experimental setups are described in App. C. ©Wētā FX

- Component multiplying $\mathbf{u}_{\mathcal{T}}^a$:

$$\Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} [m_{\mathcal{T}}^a] - \Delta t [D_{\mathcal{T}}] = \mathbf{0} \quad (50)$$

- Component multiplying p_C :

$$\begin{aligned} \Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} \Delta t \mathbf{G} + \Delta t [\phi_{\mathcal{T}}^s] \mathbf{G} &= \\ = \Delta t \left(\Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} + [\phi_{\mathcal{T}}^s] \right) \mathbf{G} &\equiv \Delta t \mathbf{F}^s \mathbf{G} \end{aligned} \quad (51)$$

- Right-hand side:

$$\begin{aligned} \Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} \left([m_{\mathcal{T}}^s] \bar{\mathbf{u}}_{\mathcal{T}}^s + [m_{\mathcal{T}}^a] \bar{\mathbf{u}}_{\mathcal{T}}^a \right) + [m_{\mathcal{T}}^s] \bar{\mathbf{u}}_{\mathcal{T}}^s &= \\ = \left(\Delta t [D_{\mathcal{T}}] [m_{\mathcal{T}}^a]^{-1} + \mathbb{I}_{\mathcal{T}} \right) [m_{\mathcal{T}}^s] \bar{\mathbf{u}}_{\mathcal{T}}^s + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^a &\equiv \\ \equiv \mathbf{P}^s \bar{\mathbf{u}}_{\mathcal{T}}^s + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^a \end{aligned} \quad (52)$$

When simplifying Eqs. 49-52 we identify the matrices $\mathbf{M}^s$, $\mathbf{F}^s$, and $\mathbf{P}^s$, as they are defined in Eqs. 37-39. Written in matrix form

$$\begin{pmatrix} \mathbf{M}^s & \mathbf{0} & \Delta t \mathbf{F}^s \mathbf{G} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\mathcal{T}}^s \\ \mathbf{u}_{\mathcal{T}}^a \\ p_C \end{pmatrix} = \mathbf{P}^s \bar{\mathbf{u}}_{\mathcal{T}}^s + \Delta t [D_{\mathcal{T}}] \bar{\mathbf{u}}_{\mathcal{T}}^a, \quad (53)$$

we see that Eq. 53 recovers the first row of Eq. 36. An analogous derivation yields the second row.

C Parameter studies

In Fig. 19 we perform parameter studies of a few parameters. Below we expand on the experimental setups.

Dispersed radius scale, ψ^r . We use the same simulation and rendering parameters as the rightmost configuration in Fig. 9b, and vary ψ^r . Since the submerged fraction is below the dispersed threshold everywhere in the simulation domain (i.e., $\phi^s < \phi^{UR}$), the decrease in effective particle size (Eq. 28) results in greater phase-coupling due to increased drag forces (Eqs. 29-30).

Saturated fraction threshold, ϕ^R . Using the beach wave scenario (Figs. 10 and 15) we show the effect of changing ϕ^R . As the fraction saturation threshold increases, the size of the fraction correction region shrinks, causing the likelihood of bubbling artifacts appearing to increase. Visually, these artifacts look similar to the ones observed when removing the fraction correction procedure (cf. Fig. 15).

Dispersed fraction threshold, ϕ^{UR} . We model a 10 m wide waterfall with varying emission fraction to investigate the effect of varying ϕ^{UR} . The waterfall emits a uniform 8 PPV, but scales particle size to model more-or-less submerged material: the emitted fraction is $\phi^s = 1$ to the left and $\phi^s = 0.001$ to the right, with a linear ramp in between. Particles are colored by size; all versions use $\phi^R = 1$. Although the visual differences between versions are small—a result of the flow being dominated by inertial forces—a smoother leading edge can be observed in regions above the dispersed threshold.

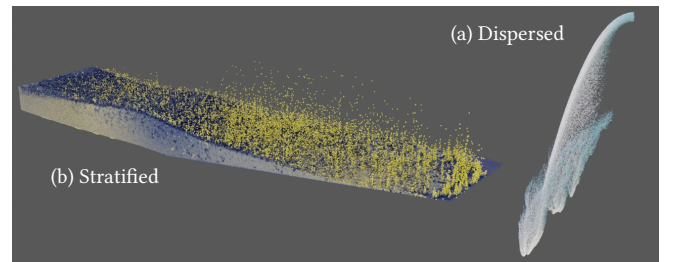


Fig. 20. *Comparison to [Wretborn et al. 2025].* We find the method of Wretborn et al. [2025] to be stable at high density ratios for dispersed water-in-air scenarios (a), but unstable for stratified interface simulations (b). Both examples were run with a density ratio 1:1000 and two Newton iterations. (a) Same initial conditions as Fig. 19a; (b) same initial conditions as Fig. 19b. ©Wētā FX

Table 3. Simulation settings and performance characteristics for all examples. ¹Resolution multiplier for the ϕ^s and ρ^s fields. ²Sparse bucket count (App. A). ³Number of simulated frames. ⁴The waterfall example was distributed over three machines; performance figures are reported for the entire simulation. ⁵These examples were executed on a cluster of various machines with 8-12 physical CPU cores.

Example	ρ [kg/m ³]	N	Λ	h [cm]	r_q [cm]	ψ^r	ϕ^s res. ¹	$ D^a ^2$	$ \mathcal{P} $	$ C $	# ³	Runtime
Fig. 1: Waterfall ⁴	1:1000	N^0	$\tilde{\Lambda}^0$	25	[1.25, 7.75]	0.01	1×	762k	277M	390M	445	18h41m
Fig. 2: Coastal cliffs	1:1000	N^1	Λ^0	30	9.31	0.01	2×	94.3k	56.9M	48.3M	600	34h21m
Fig. 3: Geyser	1:1000	N^0	$\tilde{\Lambda}^0$	2.5	[0.36, 0.78]	0.05	1×	66.6k	18.7M	34.1M	240	20h38m
Fig. 5: Bubble barrel	1000:10	N^2	Λ^0	1	0.39	1	1×	59.7k	950k	30.6M	500	9h33m
Fig. 4: Waterbombing	1:1000	N^0	$\tilde{\Lambda}^0$	2.5^{1-4}	0.39	0.05	1×	62.2k	80.9M	32.9M	240	41h24m
Fig. 9b: Simple waterfall ⁵	1:1000	N^1	Λ^0	5	1.55	0.01	1×	5.33k	790k	2.73M	192	50m
	1:1000	N^1	Λ^0	5	0.96	0.01	1×	17.3k	804k	8.86M	192	1h52m
	1:1000	N^1	Λ^0	5	0.62	0.01	1×	24.6k	897k	12.6M	192	2h0m
	1:1000	N^1	Λ^0	5	0.39	0.01	1×	18.6k	986k	9.50M	192	1h8m
	1:1000	N^1	Λ^0	5	0.25	0.01	1×	9.27k	984k	4.75M	192	38m
	1:1000	N^1	Λ^0	5	0.16	0.01	1×	4.24k	983k	2.17M	192	14m
Fig. 10: Beach wave	1:1000	N^0	$\tilde{\Lambda}^0$	5	1.55	1	1×	10.1k	8.89M	5.19M	245	52m
	1:1000	N^1	Λ^0	5	1.55	1	1×	10.1k	8.89M	5.19M	245	54m
	1:1000	N^2	Λ^0	5	1.55	1	1×	10.2k	8.89M	5.22M	245	1h13m
	1:1000	N^3	Λ^0	5	1.55	1	1×	10.2k	8.89M	5.21M	245	1h44m
	1:1000	N^0	$\tilde{\Lambda}^0$	5	1.55	1	2×	10.1k	8.89M	5.19M	245	57m
	1:1000	N^1	Λ^0	5	1.55	1	2×	10.1k	8.89M	5.18M	245	1h13m
	1:1000	N^2	Λ^0	5	1.55	1	2×	10.1k	8.89M	5.20M	245	1h16m
	1:1000	N^3	Λ^0	5	1.55	1	2×	10.2k	8.89M	5.20M	245	1h57m
Fig. 11: Barrel wave	1:1000	N^0	$\tilde{\Lambda}^0$	15	[0.75, 4.65]	0.01	1×	95.8k	200M	49.1M	400	21h21m
Fig. 12: Splitting heuristic												
Fig. 12a	1:1000	N^1	Λ^0	5	0.96	0.01	1×	89.9k	1.04M	44.5M	144	3h52m
Fig. 12b	1:1000	N^1	Λ^0	5	[0.15, 0.96]	0.01	1×	90.0k	49.6M	45.6M	144	5h22m
Fig. 12c	1:1000	N^1	Λ^0	5	[0.02, 2.0]	0.01	1×	91.3k	420k	46.7M	144	4h17m
Fig. 16: Solver comparison												
Fig. 16a	1:1000	N^1	Λ^δ	1	0.31			22.0k	64M	11.2M	241	
Fig. 16b	1:1000	N^1	Λ^δ	1	0.31			22.0k	2.96M	11.2M	241	
Fig. 16c	1:1000	N^1	Λ^δ	1	0.31			9.67k	2.96M	2.86M	241	49m
Fig. 16d	1:1000	N^1	Λ^δ	1	0.31			8.82k	2.96M	2.65M	241	44m
Fig. 16e	1:1000	N^0	$\tilde{\Lambda}^0$	1	0.31	1	1×	7.33k	2.96M	3.74M	241	1h23m
Fig. 16f	1:1000	N^0	$\tilde{\Lambda}^0$	1	0.31	1	1×	8.27k	2.96M	4.23M	241	1h16m
Fig. 14: Bubble column												
Fig. 14a	1000:1	N^0	$\tilde{\Lambda}^0$	0.25	[0.01, 0.08]	1	1×	3.22k	592k	1.65M	481	33m
Fig. 14b	1000:1	N^0	$\tilde{\Lambda}^0$	0.25	0.03	1	1×	3.09k	592k	1.58M	481	29m
Fig. 19: Parameter studies												
Fig. 19a	1:1000	N^1	Λ^0	5	0.16	[0.01, 1]	1×				192	
Fig. 19b	1:1000	N^2	Λ^0	5	1.55	1	1×				245	
Fig. 19c	1:1000	N^1	Λ^0	10	[0.0031, 0.031]	0.01	1×				72	