DOI: 10.1111/cgf.70033
EUROGRAPHICS 2025 / A. Bousseau and A. Dai
(Guest Editors)

COMPUTER GRAPHICS *forum*

A unified multi-scale method for simulating immersed bubbles

Joel Wretborn[1,2], Alexey Stomakhin[1], and Christopher Batty[2]
[1] Wētā FX
[2] University of Waterloo

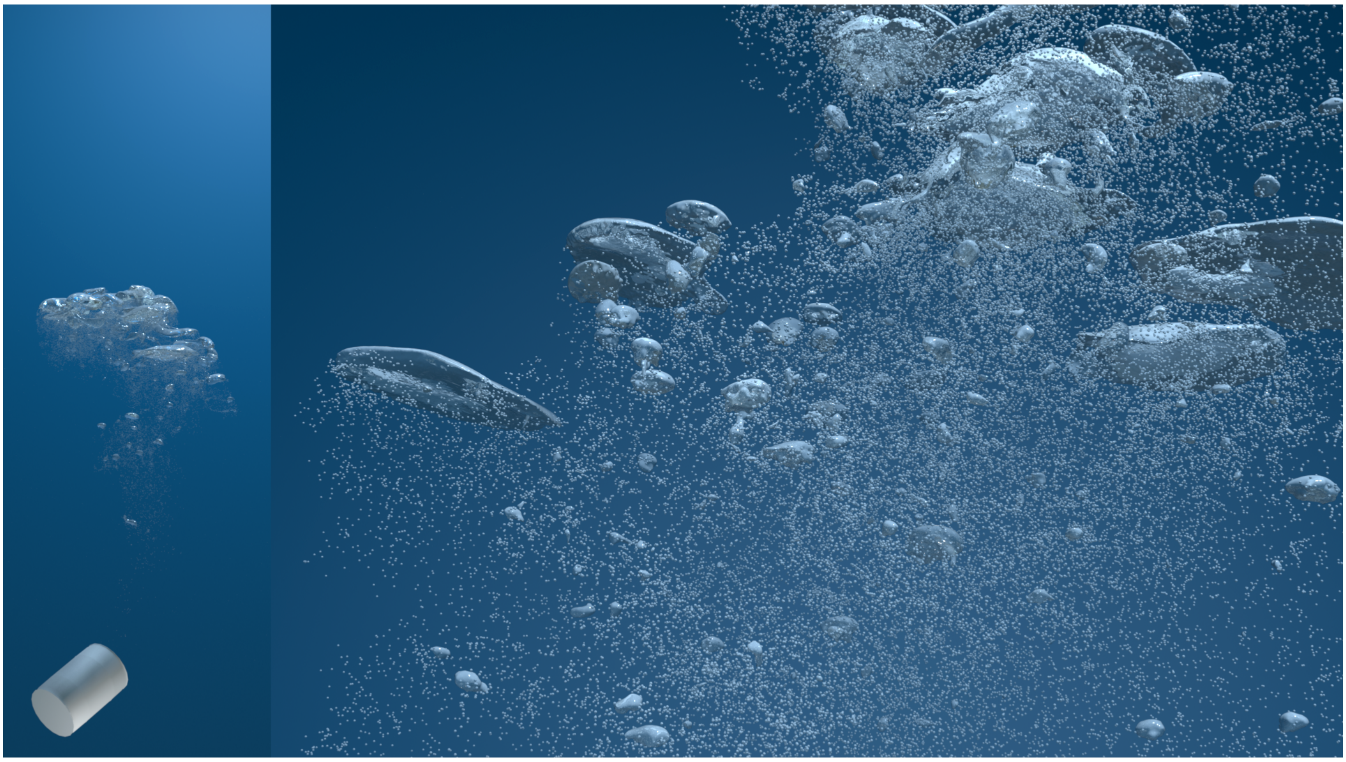

Figure 1: ***Overturning barrel.*** An inverted air-filled barrel under water is tipped upright. The resulting bubbles proceed to rise, with some collecting in large *hero* shapes and others dispersing, mixing with the water into *diffuse* regions. Our method treats this scenario with a single unified discretization, allowing bubbles to transition smoothly between the two regimes. The right image shows a zoomed-in version of the simulation on the left. ©Wētā FX

Abstract

We introduce a novel unified mixture-based method for simulating underwater bubbles across a range of bubble scales. Our approach represents bubbles as a set of Lagrangian particles that are coupled with the surrounding Eulerian water volume. When bubble particles are sparsely distributed, each particle, typically smaller than the liquid grid voxel size, corresponds to an individual spherical bubble. As the sub-grid particles increase in local density our model smoothly aggregates them, ultimately forming connected, fully aerated volumetric regions that are properly resolved by the Eulerian grid. We complement our scheme with a continuous surface tension model, defined via the gradient of the bubbles' local volume fractions, which works seamlessly across this scale transition. Our unified representation allows us to capture a wide range of effects across different scales—from tiny dispersed sub-grid air pockets to fully Eulerian two-phase interfacial flows.

CCS Concepts

• ***Computing methodologies*** → ***Physical simulation;***

1. Introduction

Underwater bubbles are a captivating natural phenomenon. From an explosive air burst from a breathing scuba diver to a swirly aerated flow in the aftermath of a crashing wave, bubble dynamics are rich with visually appealing characteristics at a multitude of scales. It is this variety of scales and unique behaviors associated with bubbles that make them so attractive to visual effects practitioners, yet so challenging to simulate numerically.

A common strategy to simulate bubble dynamics is to solve the two-phase fluid equations of motion on an Eulerian grid. In the engineering literature the liquid volume is often represented with either volume-of-fluid (VOF) (e.g., [DKP16; TSG15; KLK22]) or level set approaches [OF02; EFFM02], while the computer graphics community nowadays tends to prefer hybrid marker-based methods [BB12; GAB20; SWBD20; SLW*23] for better interface tracking at large time steps. We label this family of approaches as *hero* methods because they are widely used in visual effects for bubbles near a camera, where the dynamics of the air-water interface are of critical importance.

Visual effects artists often push grid resolutions of hero methods down to as fine as 1 mm per voxel to capture as much of the interface dynamics as possible. Although recent work has aggressively sparsified the fluid domain to simulate only where bubbles are present [SWBD20; SLW*23; FSWW24], the computational cost at such extreme resolutions still limits the total air-water volume that can be simulated at once. As a result, when there are many small dispersed bubbles scattered throughout a larger water volume, these costs quickly become prohibitive and necessitate an alternative simulation strategy. We refer to such regions as *diffuse* since the presence of these clouds of tiny bubbles tends to create a foggy appearance. Diffuse bubble flows of this type are commonly found in turbulent and aerated regions, such as those that surround a whale after a breach.

The state of the art for treating diffuse regions is to introduce massful[†] sub-grid particles representing bubbles coupled to the Eulerian water phase [KSK10; PAKF13; DB17; TGK*17; WFS22]. With such diffuse methods, satisfying results can typically be achieved with Eulerian grid resolutions on the order of centimeters per voxel, which makes it feasible to simulate large volumes of water with millions of bubbles.

Visual effect artists therefore tend to reserve hero methods for close-up camera shots where the bubbles are the focus, and diffuse methods for faraway shots. However, many scenarios fall somewhere in between and require a combination of hero and diffuse bubbles; a typical example is provided by Stomakhin, Lesser, Wretborn, et al. [SLW*23, Fig. 16]. A natural solution is to juxtapose or combine the two methods, leveraging the discretization that is most performant for the region in question and using spatial heuristics to determine which to use where. Variants of this strategy have been applied for bubbles (in water) [PAKF13; SWBD20] and spray (in air) [LTKF08; LSD*22; SLW*23]. Although the details differ, all such approaches require a sharp state change in the discrete representation when converting between the diffuse and hero fluid models. Most commonly, a passively advected marker particle (used for hero-scale interface tracking) is suddenly reinterpreted as an active, massful, diffuse particle, with the two regimes weakly coupled together [LTKF08; SWBD20; LSD*22; SLW*23]; however, strongly coupled monolithic systems have also been devised [PAKF13]. A challenge for all state-change models is to perform the transition without generating visual artifacts, which can easily happen if physical quantities are not conserved, or if renderable geometry changes.

Our goal is to develop a single unified discretization for bubble dynamics that can smoothly simulate, on one end, under-resolved diffuse bubble flows, and on the other, fully resolved hero bubble interfacial flows. We will achieve this by using massful Lagrangian particles as the primary representation for both our hero and diffuse regimes, coupled with an Eulerian grid in a modified Newton solver. Our high-level technical contributions are:

- A unified mixture model, that uses the same discrete representation for both hero and diffuse bubble regions.
- An extended Continuum Surface Force (CSF) treatment of surface tension [BKZ92] for both diffuse and hero configurations.
- A *stabilized inertia-aware* coupling scheme between Eulerian fluids and Lagrangian particles capable of simulating inviscid two-phase flows, with implicit integration of nonlinear force terms.

We achieve our unified representation by rasterizing particle properties to the grid with the Generalized Interpolation Material Point (GIMP) approach [BK*04], and we show that, through the introduction of a simple clamping function ϕ^c (§5.1), particles can be considered either as markers or massful depending on their size relative to the background grid. As a result, no distinction is made between "hero" or "diffuse" regions, and every simulated particle is treated identically within the solver. We demonstrate the generality of our mixture model and time integrator by discretizing bubbles with two different choices of basis function spaces, the

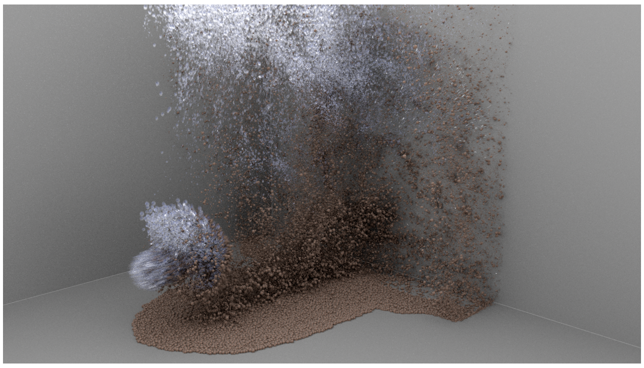

Figure 2: ***Bubble-sand mixture.*** Our model can handle varying density mixtures, here highlighted with a bubble-sand mixture pumped into a box of water. Geometric collisions between all particles and against the wall are resolved using [Dav20], ensuring there is no interpenetration. ©Wētā FX

† We will use the term "massful" here to represent particles possessing mass, distinguishing them from the massless marker particles in PIC-like schemes.

Table 1: Summary of notation used to describe our method. The discretized location (particle, cell, or face) of the quantities are provided where possible. Quantities defined for both phases are marked with a superscript *.

Notation	Units	Location	Meaning
$\mathbf{u}^\star$	m/s	q, τ	velocity
$\phi^\star$	1	$\mathbf{i}, \tau$	fraction
$\bar{\phi}^b$	1	τ	unclamped bubble fraction
ϕ^c	1	τ	fractional clamp multiplier
$\mathbb{H}$	1/m^2	$\mathbf{i}$	fraction Hessian
p	N/m^2	$\mathbf{i}$	mixture pressure
λ	1		pressure multiplier
m	kg	q	mass
r	m	q	radius
V	m^3	$q, \mathbf{i}, \tau$	discrete element volume
$\rho^\star$	kg/m^3	q, τ	density
$\mathbf{f}^\star$	N/m^3	q, τ	total force density
$\nabla\mathbf{f}^\star$	Ns/m^4	q, τ	total force density Jacobian
$\mathbf{f}^\sigma$	N/m^3	τ	surface tension force density
$\mathbf{d}^\star$	N/m^3	q, τ	drag force density
$\mathbf{D}$	N/m^3	q, τ	per-particle drag force
κ	1/m	$\mathbf{i}$	curvature
ψ	1	τ	surface tension fraction filter
ν			bubble pocket
$\mathcal{O}_\nu$			set of faces in pocket ν
$\mathbf{R}$	N		pocket residual force
χ	N/m^3		drift compensation force density
h	m		grid spacing
$r_h^{\text{surfacing}}$	m		smallest surfacing radius for given h
σ	N/m		surface tension coeffient
μ	Ns/m^2		dynamic viscosity
$\mathbf{g}$	m/s^2		gravity
Λ	1		particle interpolation function
ω	1		interpolation weights
N	1		trilinear interpolation function
χ^ε	1		power interpolation function
$\mathbf{c}$	m		power cell centroid
Ω			simulation domain
$\mathcal{P}$			set of all particles
$\mathcal{C}$			set of all cells
$\mathcal{T}$			set of all faces
$\mathcal{T}_x, \mathcal{T}_y, \mathcal{T}_z$			set of faces per component
$\mathcal{N}$		$\mathbf{i}, \tau$	set of neighboring cells/faces
$\mathbf{e}$	1	τ	face basis vector

trilinear basis and the power kernels (§5.1.1), and by recreating both hero and diffuse bubble effects. Since our coupling scheme combines an Eulerian variable-density Poisson problem and a Lagrangian ODE solver, we show how to implement it using commonly available components. Furthermore, our implicit integration of nonlinear drag forces on bubbles requires no changes to the pressure solver, in contrast to the method of Wretborn, Flynn, and Stomakhin [WFS22].

In two-phase flows, a distinction is sometimes made between miscible fluids (i.e., the mixture is homogeneous) and immiscible ones (i.e., the mixture is heterogeneous). Under this categorization, our method addresses (partially) under-resolved *immiscible* flows on an Eulerian grid. We will generally refer to this configuration as a *mixture*.

2. Related work

Although single-phase free-surface liquids are a common scenario in graphics, our interest lies in the two-phase flows of immersed bubble dynamics, which are driven by the density differences between air and water, together with surface tension effects. The discussion below emphasizes this setting.

Eulerian methods. Two-phase bubble dynamics are commonly treated using direct numerical simulation on a Eulerian grid, where the interface is discretized with either VOF [HN81; BKZ92; Pop09; KLK20] or level set methods [Kim05; KLL*07; HSKF07]. It is also possible to enhance Eulerian interface treatments with Lagrangian data, such as particles [EFFM02], or to combine Eulerian simulation with a Lagrangian interface mesh [TWGT10; BBB10; SZF12; RLZ*21].

Eulerian methods are flexible and can be used in single- and multi-phase flows, as well as (im-)miscible fluids. However, we are not aware of any purely Eulerian mixture theory-based methods that tackle the immiscible fluid flows we are interested in. A common shortcoming of Eulerian methods is that the smallest fluid feature size is limited by the grid resolution, and the computational cost quickly grows prohibitively large for higher resolutions.

Recently proposed kinetic (lattice Boltzmann) solvers can handle both miscible and immiscible fluids with exact mass and momentum conservation [LLD*21; LWD24], by leveraging a phase-field fluid model ([Kim12]) with force-based surface tension. Although outside the scope of our work, it could be interesting to explore augmenting such schemes with sub-grid bubbles.

Lagrangian methods. Smoothed particle hydrodynamics (SPH) is the most widely used Lagrangian method in graphics, and both multiphase flows [MCG03; SP08; RLY*14] and surface tension effects [IBAT11; AAT13; JWL*23] have been considered. One benefit of Lagrangian methods in the single-phase case is the ability to focus the computation on a region of interest, by only specifying particles for that region. However, specifying dynamic sparse simulation domains for two-phase flows, as in Fig. 8, is generally difficult due to the open free boundary, and we are not aware of such an SPH scheme.

Other purely Lagrangian discretizations include mesh-based boundary element schemes [DBWG15; DHB*16], tetrahedra [WTGT09; MBE*10; MEB*14], and power diagrams [dGWH*15]; however, immersed bubbles have not seen much investigation within these frameworks.

Hybrid methods. Marker-based FLIP/APIC-style methods are the standard for liquid simulations in graphics [Hou; Man; Bif; LSD*22; SLW*23]. Traditionally, they are used in single-phase scenarios, where the water surface is implied by the union of the spheres instanced at each marker particle [ZB05] (although a variety of surfacing techniques are used for final rendering). Boyd and Bridson [BB12] extended the method to two phases, where markers were added to the air phase. However, this approach introduces two descriptions of the interface that must be resolved. Stomakhin, Wretborn, Blom, and Daviet [SWBD20] circumvented this issue by only using particles to track the air phase, leaving the water as purely Eulerian; subsequent authors followed suit [SLW*23; FSWW24].

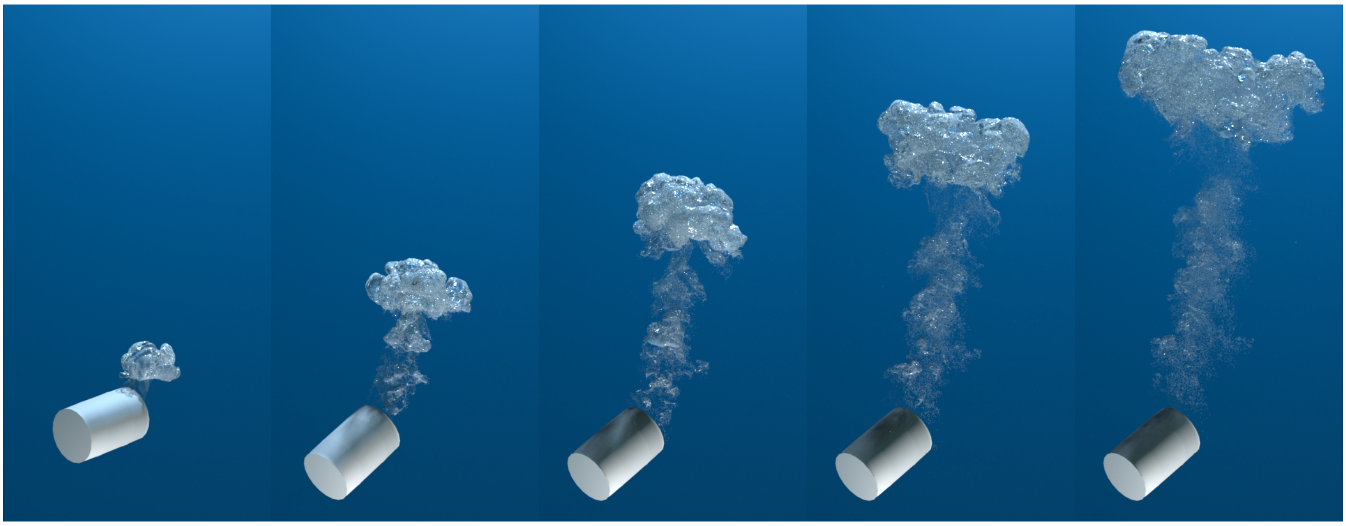

Figure 3: ***Barrel timelapse.*** An upside down air-filled barrel is overturned to release bubbles. The hydrostatic boundary condition on the simulation domain boundary (§5.5) keeps the air in the container under gravity until a critical angle is reached and the air spills out. ©Wētā FX

Existing hybrid massful methods tend to assume a two-phase mixture, where massful particles are immersed in an otherwise Eulerian fluid. These methods employ either VOF schemes, by rasterizing volumetric fractions [AO96; KSK10; WFS22], or the material point method (MPM), with the fluid(-like) material treated as weakly compressible [DB17; TGK*17]. Our proposed method fits into the VOF category.

Multiple discretizations. We have already mentioned several methods that combine multiple discretizations into a single solver [PAKF13; LTKF08; SWBD20; LSD*22]. Ren, Jiang, Li, and Lin [RJLL15] combine a VOF simulator with a post-process particle-based bubbles simulation. Outside of the bubbles context, another example of such a combination is the work of Fei, Batty, Grinspun, and Zheng [FBGZ18], who combine a 2D reduced flow model for wet cloth with a 3D particle representation for drips.

3. Model

We model submerged bubbles as an incompressible two-phase mixture of air and water. We define $\phi^b : \Omega \to [0,1]$ as the (unitless) fractional amount, or simply *fraction*, of air at any point in the simulation domain $\Omega \subseteq \mathbb{R}^3$, and the complement $\phi^w(\mathbf{x}) = 1 - \phi^b(\mathbf{x})$ as the fractional amount of water. For equations of motion we use the two-fluid mixture model [WFS22]

$$\phi^b \rho^b \frac{D^b \mathbf{u}^b}{Dt} = \phi^b \rho^b \mathbf{g} + \mathbf{d}^b - \phi^b \nabla p, \tag{1}$$

$$\phi^w \rho^w \frac{D^w \mathbf{u}^w}{Dt} = \phi^w \rho^w \mathbf{g} + \mathbf{d}^w - \phi^w \nabla p, \tag{2}$$

$$0 = \nabla \cdot (\phi^b \mathbf{u}^b + \phi^w \mathbf{u}^w), \tag{3}$$

for velocity $\mathbf{u}$, density ρ, pressure p, drag force density $\mathbf{d}$, external accelerations $\mathbf{g}$ (e.g. gravity), and material derivative operator $\frac{D}{Dt}$ for the respective phase. Superscripts b or w indicate that a quantity is defined for the air (bubble) or water phase, respectively. In particular, the superscript on the material derivative indicates the velocity field through which the quantity is advected, e.g., $\frac{D^w}{Dt}$ implies the operator $\frac{\partial}{\partial t} + \mathbf{u}^w \cdot \nabla$. To ensure momentum conservation of the drag force, we set $\mathbf{d}^w = -\mathbf{d}^b$. In addition to Eqs. 1-3 there are also conservation laws for densities and fractional amounts, discussed in the App. A, which will be automatically ensured by our discretization of bubbles as massful particles (§5).

Equations 1-2 couple the two phases via drag forces and a generalized buoyancy term [WFS22; DB17]. The dynamics in our case will be primarily driven by the density difference of the two fluids, along with surface tension which we discuss in §3.1. We will treat the pressure term with a projection strategy similar to that of a standard single-phase pressure projection (§5.5). The drag force $\mathbf{d}^b$ is defined in §5.2 once the discretization has been presented.

A key feature of diffuse bubble methods is that they track the fluid mixture using one or more continuous indicator variables, whereas interfacial flow models all rely on tracking a sharp interface (by e.g., level set, reconstruction from particles, etc.). Nevertheless, they are closely related. Consider an interfacial liquid with $\phi^b \in \{0,1\}$ (i.e., a strict indicator function) on the entire domain, and for the moment set the drag force $\mathbf{d}^b = \mathbf{d}^w = 0$. Assuming appropriate boundary conditions, the only difference between Eqs. 1-3 and interfacial flow models is that the latter typically includes a surface tension force. Therefore, to pursue our goal of unifying these distinct settings, we next propose a consistent surface tension model that can be applied to continuous fluid mixtures.

3.1. A continuum surface tension model for bubble dynamics across scales

Surface tension acts to minimize the area of the surface between two fluids, and can be modeled as a pressure jump $[p]$ across the interface proportional to mean curvature $\kappa(\mathbf{x})$ with coefficient σ. Since such a jump is mathematically modelled as a singular source term (i.e., Dirac delta) supported only on the interface, Brackbill, Kothe, and Zemach [BKZ92] proposed a smoothed approximation where the pressure jump across a small element A of the interface is replaced by a volumetric force density field $\mathbf{f}^\sigma$ acting on a region

$V(A)$ around that element

$$\int_A [p]\mathbf{n}dS = \int_A \sigma\kappa\mathbf{n}dS \approx \int_{V(A)} \mathbf{f}^{\sigma} dV.$$

This approach allowed them to simulate surface tension effects while sidestepping the numerical challenges of sharp discontinuities. From a binary indicator variable $\tilde{\phi} : \Omega \to \{0,1\}$, where $\tilde{\phi} = 0$ in one phase and $\tilde{\phi} = 1$ in the other, they apply convolution to create a smoothed variable $\phi : \Omega \to [0,1]$, from which they in turn define the volumetric surface tension force density

$$\mathbf{f}^{\sigma} = \sigma\kappa(\mathbf{x})\nabla\phi(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega. \tag{4}$$

Because ϕ is constructed as a smoothed version of $\tilde{\phi}$, it implicitly describes an interface located at the $\phi = 0.5$ isosurface. Thus, while Eq. 4 is continuous in formulation, the manner in which Brackbill, Kothe, and Zemach [BKZ92] employed it falls squarely in the interfacial flow category. We will next adapt this model to our diffuse mixture setting.

We take as our fundamental fluid description a collection of small dispersed bubbles, each of which is assumed to be in equilibrium with respect to local surface tension forces (and thus is spherical). These bubbles, represented as particles, will be used to determine our bubble fractions ϕ^b; for details see §5.1, but for the proceeding discussion it is enough to know that ϕ^b is a fractional field that has a higher value where there is more bubble volume present. A pure air phase corresponds to $\phi^b = 1$.

Multiple bubbles near each other would collect and form new interface shapes, but these are generally too complex and too numerous for us to compute individually. Instead, we *assume* the presence of an interface wherever there is a nonzero volume fraction gradient $\nabla\phi^b(\mathbf{x})$ and compute the force density

$$\mathbf{f}^{\sigma} = \sigma\kappa(\mathbf{x})\nabla\phi^b(\mathbf{x}), \quad \kappa(\mathbf{x}) = \nabla \cdot \left(\frac{\nabla\phi^b(\mathbf{x})}{||\nabla\phi^b(\mathbf{x})||} \right). \tag{5}$$

Equation 5 is, in form, identical to the formulation of Brackbill, Kothe, and Zemach [BKZ92] but with an important distinction: ϕ^b does not track an explicit interface. For Brackbill, Kothe, and Zemach [BKZ92], ϕ is a smooth field constructed from some sharp interface representation, whereas for us ϕ^b is a volume fraction representation of dispersed bubbles immersed in a liquid. As a result, the set of possible functions $\{\phi^b\}$ is a superset of $\{\phi\}$. Then, $\mathbf{f}^{\sigma}$ in Eq. 5 acts on the bubble-water *mixture* rather than the two phases separately, and since

$$\mathbf{f}^{\sigma} = (\phi^b + \phi^w)\mathbf{f}^{\sigma} = \phi^b\mathbf{f}^{\sigma} + \phi^w\mathbf{f}^{\sigma}, \tag{6}$$

a natural choice is to apply the fraction-scaled surface tension force density to each phase.

We finally arrive at our equations of motion by combining Eqs. 1-2 with Eq. 5, resulting in

$$\phi^b\rho^b\frac{D^b\mathbf{u}^b}{Dt} = \phi^b\rho^b\mathbf{g} + \mathbf{d}^b - \phi^b\nabla p + \phi^b\mathbf{f}^{\sigma}, \tag{7}$$

$$\phi^w\rho^w\frac{D^w\mathbf{u}^w}{Dt} = \phi^w\rho^w\mathbf{g} + \mathbf{d}^w - \phi^w\nabla p + \phi^w\mathbf{f}^{\sigma}, \tag{8}$$

$$0 = \nabla \cdot (\phi^b\mathbf{u}^b + \phi^w\mathbf{u}^w). \tag{9}$$

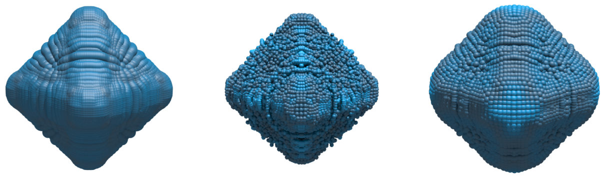

Figure 4: ***Surface tension methods.*** We compare our surface tension model to Boyd and Bridson [BB12]. An initial cube seeded with regular particles oscillates to become an octahedron under surface tension. Particles are colored according to their speed, where lighter hues of blue indicate faster speeds. ©Wētā FX

Equations 7-9 are a continuous two-phase mixture model with surface tension, but without an explicit surface representation. In the case of a sharp interface $\phi^b \equiv \phi$ our surface tension model is identical to that of Brackbill, Kothe, and Zemach [BKZ92].

We can show from the equations of motion that Eq. 6 is the only sensible way to split the surface tension force contribution between phases, as it guarantees the possibility of a static volumetric equilibrium. Indeed, setting $\mathbf{u}^w = \mathbf{u}^b = \mathbf{g} = \mathbf{d}^b = \mathbf{0}$ in Eqs. 7-9 yields

$$\mathbf{0} = -\phi^b\nabla p + \phi^b\mathbf{f}^{\sigma},$$
$$\mathbf{0} = -\phi^w\nabla p + \phi^w\mathbf{f}^{\sigma},$$

where both can be satisfied when $\nabla p = \mathbf{f}^{\sigma}$.

We finally point out that our method is not too dissimilar from existing SPH surface tension models that use color fields (e.g. [MCG03]).

4. Temporal discretization

We will solve the equations of motion using operator splitting and a modified Newton-Raphson loop that separates the position and velocity updates. For the proceeding section, we will use $\star \in \{b, w\}$ to refer to the two fluids at the same time. We first update velocities according to the fluid forces

$$\phi^\star\rho^\star\frac{\partial\mathbf{u}^\star}{\partial t} = \phi^\star\rho^\star\mathbf{g} + \mathbf{d}^\star - \phi^\star\nabla p + \phi^\star\mathbf{f}^{\sigma}, \tag{10}$$

subject to Eq. 9 and then separately solve the advection equation for each of the two phases,

$$\frac{D^\star\mathbf{u}^\star}{Dt} = \mathbf{0}.$$

Below, we consider the first set of equations and defer discussion of advection to §5.6 after our spatial discretization has been introduced.

4.1. Full Newton method

Let $n = 0, 1, \ldots$ denote the iteration index and consider a forward simulation time step Δt. We use a subscript n to represent a quantity at that Newton iteration, and use $n = 0$ to indicate the input values before the first Newton iteration.

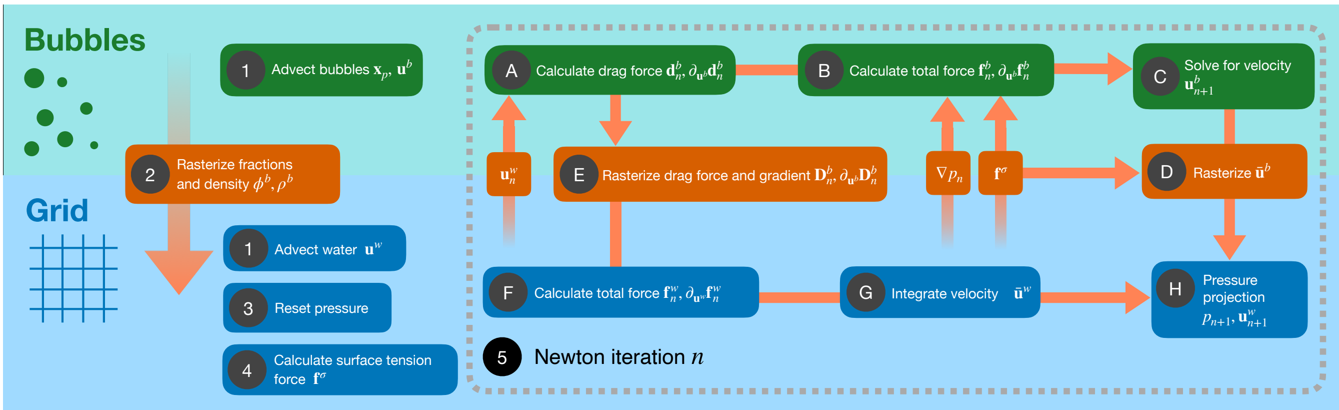

Figure 5: ***Algorithm outline.*** We provide a visual description of the steps in Alg. 1. Green boxes correspond to operations performed on the particles, blue on the grid, and orange require either interpolation or rasterization. Arrows indicate the flow of data. ©Wētā FX

Discretizing Eq. 10 in time yields

$$\phi^\star \rho^\star \left(\frac{\mathbf{u}^\star - \mathbf{u}_0^\star}{\Delta t} \right) = \phi^\star \rho^\star \mathbf{g} + \mathbf{d}^\star - \phi^\star \nabla p + \phi^\star \mathbf{f}^\sigma.$$

We freeze attributes related to positions, holding ϕ and ρ constant for all iterations. We seek the roots $\mathbf{f}^\star = \mathbf{0}$ of the per-phase force density

$$\mathbf{f}^\star(\mathbf{u}^b, \mathbf{u}^w, \lambda) = \phi^\star \rho^\star \mathbf{g} + \mathbf{d}^\star - \lambda \phi^\star \nabla p + \phi^\star \mathbf{f}^\sigma + \phi^\star \rho^\star \left(\frac{\mathbf{u}_0^\star - \mathbf{u}^\star}{\Delta t} \right), \tag{11}$$

subject to Eq. 9. The argument $\lambda \in \{0, 1\}$ has been introduced to allow for the removal of the pressure gradient term ($\lambda = 0$), which will be convenient for the modified Newton solve we describe later. Unless stated otherwise, however, we assume $\lambda = 1$. The corresponding Jacobian $\nabla \mathbf{f}^\star : (\mathbf{u}^b, \mathbf{u}^w) \to \mathbb{R}^{6\times 3}$ has components

$$\partial_{\mathbf{u}^\xi} \mathbf{f}^\star = \partial_{\mathbf{u}^\xi} \mathbf{d}^\star - \delta^{(\xi,\star)} \frac{\phi^\star \rho^\star}{\Delta t} \mathbb{I}, \quad \xi \in [w, b] \tag{12}$$

where we use the shorthand $\partial_{\mathbf{u}} \equiv \partial / \partial \mathbf{u}$, the Dirac delta function $\delta^{(\cdot,\cdot)}$, and $\mathbb{I}$ as the 3×3 identity matrix. Since position-related quantities are frozen, and surface tension and pressure are only indirectly affected by velocity changes, they do not contribute to the Jacobians.

From the above, we can assemble a Newton iteration as

$$-\begin{pmatrix} \partial_{\mathbf{u}^b} \mathbf{f}^b & \partial_{\mathbf{u}^w} \mathbf{f}^b \\ \partial_{\mathbf{u}^b} \mathbf{f}^w & \partial_{\mathbf{u}^w} \mathbf{f}^w \end{pmatrix} \begin{pmatrix} \mathbf{u}_{n+1}^b - \mathbf{u}_n^b \\ \mathbf{u}_{n+1}^w - \mathbf{u}_n^w \end{pmatrix} = \begin{pmatrix} \mathbf{f}^b \\ \mathbf{f}^w \end{pmatrix}, \tag{13}$$

$$\nabla \cdot (\phi^b \mathbf{u}_{n+1}^b + \phi^w \mathbf{u}_{n+1}^w) = 0. \tag{14}$$

4.2. Modified Newton solver

Instead of solving Eqs. 13-14 directly, we leverage a type of *inertia-aware* coupling scheme, introduced by Wretborn, Flynn, and Stomakhin [WFS22]. We will first review their method using our notation.

Inertia-aware, [WFS22]. Wretborn, Flynn, and Stomakhin [WFS22] propose finding new velocities $\mathbf{u}_{n+1}^b$ and $\mathbf{u}_{n+1}^w$, by employing operator splitting. By setting cross-derivative terms to zero $\partial_{\mathbf{u}^w} \mathbf{f}^b = \partial_{\mathbf{u}^b} \mathbf{f}^w = \mathbf{0}$, each phase can be solved for independently in Eq. 13. They perform the following steps:

1. Solve implicitly for $\mathbf{u}_{n+1}^b$, using pressure p_n from the last Newton iteration:
$$-\partial_{\mathbf{u}^b} \mathbf{f}^b(\mathbf{u}_n^b, \mathbf{u}_n^w) \left(\mathbf{u}_{n+1}^b - \mathbf{u}_n^b \right) = \mathbf{f}_n^b(\mathbf{u}_n^b, \mathbf{u}_n^w, \lambda \equiv 1). \tag{15}$$
2. Compute pre-pressure bubble velocity $\bar{\mathbf{u}}^b$ by canceling the (explicit) pressure force contribution added by Eq. 15:
$$\bar{\mathbf{u}}^b = \mathbf{u}_{n+1}^b + \frac{\Delta t}{\rho^w} \nabla p_n. \tag{16}$$
3. Solve explicitly for pre-pressure water velocity $\bar{\mathbf{u}}^w$:
$$\bar{\mathbf{u}}^w = \mathbf{u}_n^w + \frac{\Delta t}{\phi^w \rho^w} \mathbf{f}^w(\mathbf{u}_n^b, \mathbf{u}_n^w, \lambda \equiv 0). \tag{17}$$
At this point $\bar{\mathbf{u}}^b$ and $\bar{\mathbf{u}}^w$ have had all forces integrated except for pressure.
4. Solve implicitly for mixture pressure p_{n+1}, water velocity $\mathbf{u}_{n+1}^w$, and bubble velocity $\hat{\mathbf{u}}^b$:
$$\frac{\phi^b \rho^b}{\Delta t} \left(\hat{\mathbf{u}}^b - \bar{\mathbf{u}}^b \right) = -\phi^b \nabla p_{n+1}, \tag{18}$$
$$-\partial_{\mathbf{u}^w} \mathbf{f}^w(\mathbf{u}_n^b, \mathbf{u}_n^w) \left(\mathbf{u}_{n+1}^w - \bar{\mathbf{u}}^w \right) = -\phi^w \nabla p_{n+1}, \tag{19}$$
$$0 = \nabla \cdot (\phi^b \hat{\mathbf{u}}^b + \phi^w \mathbf{u}_{n+1}^w). \tag{20}$$
Notably, this method discards the velocity $\hat{\mathbf{u}}^b$, retaining $\mathbf{u}_{n+1}^b$ instead, and applies the pressure p_{n+1} to the bubble phase in the next Newton iteration via Eq. 15 in step (1).

Wretborn, Flynn, and Stomakhin [WFS22] observed that removing Eq. 18 and using $\mathbf{u}_{n+1}^b$ for bubble velocities treats bubbles as infinitely heavy and causes instabilities. They coined the term *inertia-unaware* to describe this variation.

The inclusion of the drag force gradient on the left-hand side in Eq. 19 effectively modifies the density of the water in the pressure solve. As a result, we found that achieving a hydrostatic equilibrium for two equal-density fluids and a nonlinear drag force is not possible for the *inertia-aware* model in the form proposed by Wretborn, Flynn, and Stomakhin [WFS22].

Stabilized inertia-aware. To address this issue we integrate water velocities implicitly in step (3) rather than in the pressure solve of Eq. 19. Consider the following steps:

1. Solve implicitly for $\mathbf{u}^b_{n+1}$ (same as *inertia-aware, [WFS22]*):

$$-\partial_{\mathbf{u}^b}\mathbf{f}^b(\mathbf{u}^b_n,\mathbf{u}^w_n)\left(\mathbf{u}^b_{n+1}-\mathbf{u}^b_n\right)=\mathbf{f}^b_n(\mathbf{u}^b_n,\mathbf{u}^w_n,\lambda\equiv 1). \tag{21}$$

2. Compute pre-pressure bubble velocity $\bar{\mathbf{u}}^b$ (same as *inertia-aware, [WFS22]*):

$$\bar{\mathbf{u}}^b=\mathbf{u}^b_{n+1}+\frac{\Delta t}{\rho^w}\nabla p_n. \tag{22}$$

3. Solve implicitly for pre-pressure water velocity $\bar{\mathbf{u}}^w$:

$$-\partial_{\mathbf{u}^w}\mathbf{f}^w(\mathbf{u}^b_n,\mathbf{u}^w_n)\left(\bar{\mathbf{u}}^w-\mathbf{u}^w_n\right)=\mathbf{f}^w(\mathbf{u}^b_n,\mathbf{u}^w_n,\lambda\equiv 0). \tag{23}$$

4. Solve implicitly for mixture pressure p_{n+1} and velocities $\mathbf{u}^w_{n+1}$, $\hat{\mathbf{u}}^b$:

$$\frac{\phi^b\rho^b}{\Delta t}\left(\hat{\mathbf{u}}^b-\bar{\mathbf{u}}^b\right)=-\phi^b\nabla p_{n+1}, \tag{24}$$

$$\frac{\phi^w\rho^w}{\Delta t}\left(\mathbf{u}^w_{n+1}-\bar{\mathbf{u}}^w\right)=-\phi^w\nabla p_{n+1}, \tag{25}$$

$$0=\nabla\cdot(\phi^b\hat{\mathbf{u}}^b+\phi^w\mathbf{u}^w_{n+1}). \tag{26}$$

As will be seen in the next section, $\mathbf{f}^w, \partial_{\mathbf{u}^w}\mathbf{f}^w$, and $\mathbf{u}^w$ are all stored at the same (face) locations on the staggered grid and the update in Eq. 23 can be done for each face independently. By contrast, the method of Wretborn, Flynn, and Stomakhin [WFS22] requires $\partial_{\mathbf{u}^w}\mathbf{f}^w$ to be computed on cell centers (Eq. 19), thereby blurring it.

One major benefit of both *inertia-aware* schemes is that they avoid the costly fully monolithic solvers favored by prior work [PAKF13; FBGZ18], and integrating them into existing solvers requires little effort. To solve Eq. 21 we use an industry standard FEM solver [LSD*22], which has the added benefit of allowing us to use the same coupling scheme for thin or permeable elastic materials such as hair and cloth (see [SWBD20; ZSI*23]). Solving Eqs 24-26 requires only a few adjustments of a single-phase incompressible pressure solver. We will elaborate on these solvers in §5.2 and §5.5.

All our examples use the *stabilized inertia-aware* coupling scheme. We provide a convergence plot in Fig. 6 showing that the two inertia-aware integration schemes provide the same characteristic behavior, although empirically we have observed improved stability using our technique.

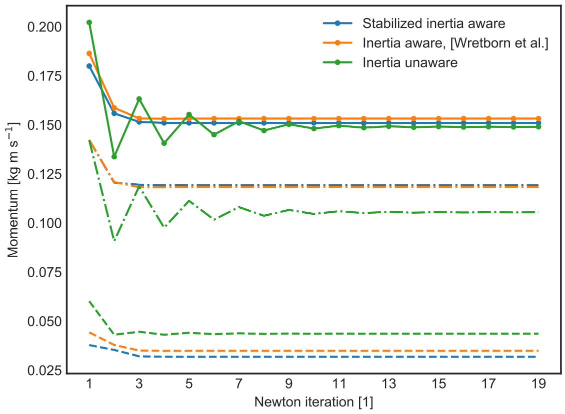

Figure 6: ***Convergence plot.*** A single moving bubble is submerged in still water, and the total (solid), particle (point-dashed), and fluid (dashed) momenta are plotted for every Newton iteration under different coupling schemes. ©Wētā FX

5. Spatial discretization

Our mixture model leverages quantities stored both on particles and on a staggered regular grid, with the latter allocated sparsely around particles in so-called *tiles* [LSD*22]. Each tile consists of a user-defined number of voxels, typically 4^3-8^3. We use the term "padding" to refer to the number of whole tiles between any particle and the ambient (not tiled) space. See Fig. 12 for an illustration of 4^3 voxels per tile with a padding of zero.

Notation. We will use $\mathbf{i}\equiv(i,j,k)\in\mathcal{C}$ for $\mathcal{C}\subset\mathbb{Z}^3$ to index the set of active *cells*[‡] of a uniform grid with spacing h, where the cell $\mathbf{i}$ is located at $\mathbf{x_i}=h\mathbf{i}$. Faces are located at a $\frac{1}{2}$-index offset in each direction, and we define $\mathcal{T}_x=\{(i-\frac{1}{2},j,k)\mid(i,j,k)\in\mathcal{C}\}$ and so on for $\mathcal{T}_y$ and $\mathcal{T}_z$. Often it is convenient to refer to all faces collectively, for which we define $\mathcal{T}=\mathcal{T}_x\cup\mathcal{T}_y\cup\mathcal{T}_z$. With each face $\tau\in\mathcal{T}$ we associate a surrounding cubic control volume $V_\tau=h^3$ and Euclidean basis vector $\mathbf{e}_\tau$ corresponding to the face normal. We define the neighbor sets $\mathcal{N}_\mathbf{i},\mathcal{N}_\tau$ to indicate the six faces touching cell $\mathbf{i}$ and the two cells touching face τ, respectively. For particles, we let $q\in\mathcal{P}$ be a running index with $\mathcal{P}\subset\mathbb{N}$.

For the rest of this section we let a subscript denote a (spatial) discretization location. If not otherwise stated, q, $\mathbf{i}$, and τ will denote a particle, a cell, and a face, respectively. Note that some quantities, such as $\mathbf{u}^b$, are discretized both on particles and on the grid. For reference, Table 1 lists important quantities and their discretized locations when possible.

5.1. Material particles and a background grid representation

We treat bubbles as massful particles with density ρ^b_q, volume V_q, and velocity $\mathbf{u}^b_q$. With each particle we associate an interpolation function $\Lambda_q:\Omega\rightarrow\mathbb{R}^+$ such that

$$\int_\Omega\Lambda_q(\mathbf{x})d\mathbf{x}=V_q. \tag{27}$$

Then, we follow the GIMP framework [BK*04; GTJS17] and define the interpolation weights

$$\omega_{q\alpha}=\frac{1}{V_q}\int_\Omega\Lambda_q(\mathbf{x})N_\alpha(\mathbf{x})d\mathbf{x},\quad\alpha\in\{\mathcal{C}\cup\mathcal{T}\}, \tag{28}$$

[‡] We will use the terms "cell" and "face" for locations within a voxel, the former being at the center of the voxel and the latter at the centers of the walls of the voxel.

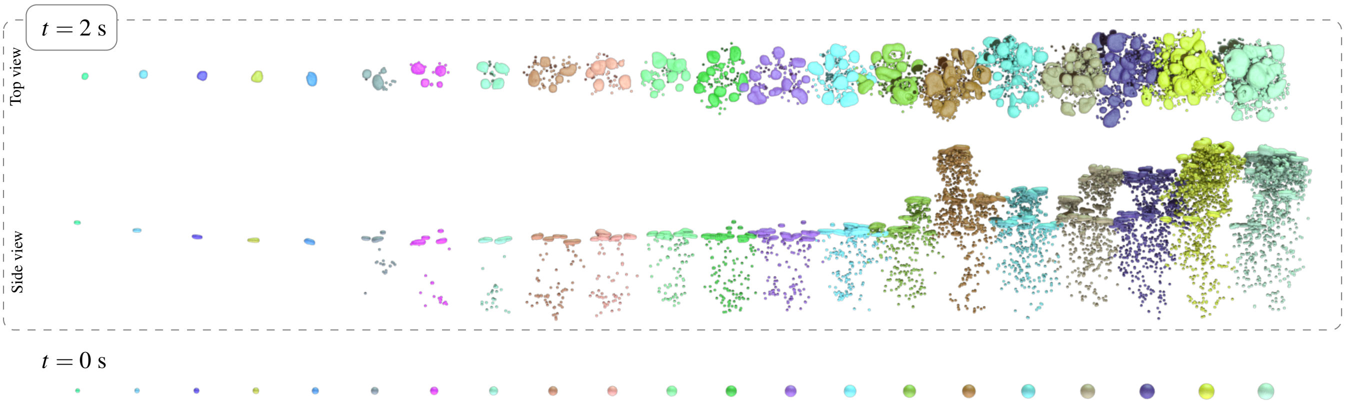

Figure 7: ***Bubble sizes.*** We compare the resulting bubble shapes from initially still air, with constant surface tension $\sigma = 0.0072$ N/m and voxel size $h = 0.125$ cm. Small bubbles (left) collect into axisymmetric shapes and rise in straight paths. As bubbles increase in size (middle) they become asymmetric and start to break up, rising in characteristically "wobbly" paths. For large bubbles (right) the bubble motion and break-up becomes chaotic. Particles are sampled every $h/2$ within a sphere of radius 3 mm to 23 mm in 1 mm increments. Particles are drawn with $4r_q$ for better visibility. ©Wētā FX

where

$$N_\alpha(\mathbf{x}) = N(\Delta\mathbf{x}_\alpha \cdot \mathbf{e}_x, h)\, N(\Delta\mathbf{x}_\alpha \cdot \mathbf{e}_y, h)\, N(\Delta\mathbf{x}_\alpha \cdot \mathbf{e}_z, h),$$

with the standard linear "hat" function $N(x,h) = \max(1 - \frac{|x|}{h}, 0)$, $\Delta\mathbf{x}_\alpha = \mathbf{x} - \mathbf{x}_\alpha$, and α the discrete element in question. The functions Λ_q must be defined such that the weights interpolate linear functions exactly:

$$\mathbf{x}_q = \sum_{i \in \mathcal{C}} \omega_{qi} \mathbf{x}_i = \sum_{\tau \in \mathcal{T}_\xi} \omega_{q\tau} \mathbf{x}_\tau, \quad \forall \xi \in \{x, y, z\},\ \forall q \in \mathcal{P}. \tag{29}$$

The interpolation functions will be used to transfer particle values to and from the grid, for which we define the following rasterization and interpolation procedures.

Rasterization. In order to construct discrete volume fractions on the grid we start by defining an unmodified *bubble fraction* $\bar{\phi}^b$ as the weighted and normalized bubble volume rasterized to faces $\mathcal{T}$:

$$\bar{\phi}^b_\tau = \frac{1}{V_\tau} \sum_{q \in \mathcal{P}} V_q \omega_{q\tau}. \tag{30}$$

Since $\bar{\phi}^b_\tau$ is a variable on faces, the rasterization above contains three rasterization procedures, one per staggered grid component. Notably, although $\bar{\phi}^b_\tau$ is stored on faces it is a scalar field.

Under our modeling assumption of an incompressible mixture a fractional value $\bar{\phi}^b_\tau > 1$ is unacceptable. To avoid this situation we introduce a fraction clamp factor $\phi^c_\tau = 1/\bar{\phi}^b_\tau$ if $\bar{\phi}^b_\tau > 1$ (otherwise letting $\phi^c_\tau = 1$), and define the clamped bubble fraction as

$$\phi^b_\tau = \phi^c_\tau \bar{\phi}^b_\tau. \tag{31}$$

The discrete water fraction is the complement $\phi^w_\tau = 1 - \phi^b_\tau$.

We define the *bubble density* ρ^b_τ for a face as its rasterized bubble mass divided by its "effective" bubble volume $\bar{\phi}^b_\tau V_\tau$ (i.e., without clamping) rasterized to $\mathcal{T}$ as

$$\rho^b_\tau = \frac{1}{\bar{\phi}^b_\tau V_\tau} \sum_{q \in \mathcal{P}} \rho^b_q V_q \omega_{q\tau}. \tag{32}$$

For $\bar{\phi}^b_\tau > 1$, a condition commonly encountered due to numerical integration errors or choice of initial conditions, we have found the treatment above to be a reasonable compromise between mild violations of mass conservation and unnatural sinking of artificially compressed bubble regions. Note that this choice means that for a constant bubble density the rasterized density will also have the same constant value.

We compute the *bubble velocity* on $\mathcal{T}$ in a momentum-conserving fashion by rasterizing momentum and dividing off bubble mass

$$u^b_\tau = \frac{\sum_{q \in \mathcal{P}} \rho^b_q V_q (\mathbf{u}^b_q \cdot \mathbf{e}_\tau) \omega_{q\tau}}{\sum_{q \in \mathcal{P}} \rho^b_q V_q \omega_{q\tau}}. \tag{33}$$

Interpolation. We interpolate quantities from $\mathcal{T}$ to $\mathcal{P}$ using

$$\mathbf{Q}_q = \sum_{\tau \in \mathcal{T}} Q_\tau \mathbf{e}_\tau \omega_{q\tau}. \tag{34}$$

5.1.1. Choice of basis functions

We have tested our method using both trilinear basis functions and the power kernel of Qu, Li, De Goes, and Jiang [QLDJ22].

Trilinear. Set $\Lambda_q(\mathbf{x}) = V_q \delta(\mathbf{x} - \mathbf{x}_q)$. This choice of the basis function recovers the trilinear weights from the diffuse bubble method outlined by Wretborn, Flynn, and Stomakhin [WFS22].

Power. Following Qu, Li, De Goes, and Jiang [QLDJ22], we can use the *power kernel* given by

$$\chi^\varepsilon_q(\mathbf{x}) = \frac{s_q e^{-||\mathbf{x}_q - \mathbf{x}||^2/\varepsilon}}{\sum_{b \in \mathcal{P}} s_b e^{-||\mathbf{x}_q - \mathbf{x}_b||^2/\varepsilon}}$$

for regularized scaling variables s_q and a user defined regularization parameter ε. We compute s_q exactly as described by Qu, Li, De Goes, and Jiang [QLDJ22], and then let $\Lambda_q(\mathbf{x}) = \chi^\varepsilon_q(\mathbf{x})$. We used the following settings related to power diagram generation for all of our examples: the s-grid resolution was defined as the Eulerian

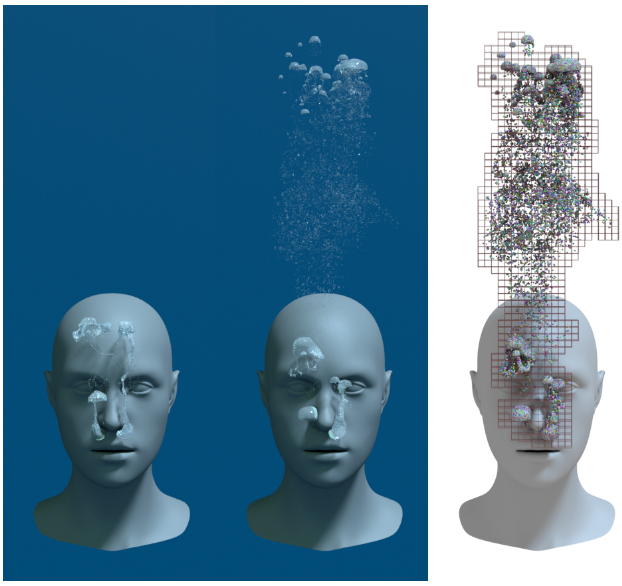

Figure 8: ***Exhale.*** A person under water exhales in bursts, resulting in characteristic bubble shapes. At the right is a diagnostic view with a slice of the sparse volumetric tiles together with underlying particles, the latter being artificially drawn with radius $2r_q$ for better visibility. ©Wētā FX

grid resolution h, t-grid resolution $h/2$, regularization amount $\varepsilon = (h/3)^2$, and kernel cut-off $3h/2$.

The set of all functions $\{\chi_q^\varepsilon(\mathbf{x})\}_{p\in\mathcal{P}}$ forms a regularized power diagram with particles as sites. Each power cell represents the same volume as that of the corresponding particle. This fact ensures that the unclamped fraction in Eq. 30 will be in the range $\bar{\phi}_\tau^b \in [0,1]$. Therefore, under the power kernel, there is no need for fraction clamping and we can simply set $\phi_\tau^b = \bar{\phi}_\tau^b$. Additionally, the resulting weights in Eq. 28 are not interpolants, but rather return the centroid $\mathbf{c}_q = \sum_{\tau\in\mathcal{T}} \omega_{q\tau}\mathbf{x}_\tau$. Thus, after computing χ_q^ε we move particle positions to the centroids $\mathbf{x}_q \leftarrow \mathbf{c}_q$, in accordance with the centroid advection scheme proposed by Qu, Li, De Goes, and Jiang [QLDJ22].

5.2. Bubbles equations of motion

Equipped with the spatial interpolation functions in the previous section, we can now discretize Eq. 21. Dividing Eqs. 11 by ϕ^b and taking density, velocity, and the drag force constant over the bubble, the total exerted force per unit of bubble volume $\mathbf{f}_q^b \equiv (\mathbf{f}^b/\phi^b)|_{\mathbf{x}_q}$ and its gradient $\partial_{\mathbf{u}^b}\mathbf{f}_q^b \equiv (\partial_{\mathbf{u}^b}\mathbf{f}^b/\phi^b)|_{\mathbf{x}_q}$ acting on each particle are given by

$$\begin{aligned} \mathbf{f}_q^b &= \rho_q^b\mathbf{g} + \mathbf{d}_q^b - \nabla p(\mathbf{x}_q) + \mathbf{f}^\sigma(\mathbf{x}_q) + \rho_q^b\left(\frac{\mathbf{u}_{0,q}^b - \mathbf{u}_q^b}{\Delta t}\right), \\ \partial_{\mathbf{u}^b}\mathbf{f}_q^b &= \partial_{\mathbf{u}^b}\mathbf{d}_q^b - \frac{\rho_q^b}{\Delta t}\mathbb{I}, \end{aligned} \tag{35}$$

where $\mathbf{u}_{0,q}^b$ represents the particle velocity at the beginning of the Newton iteration and $\mathbf{d}_q^b$ will be defined in Eq. 37. To compute the total force per particle one must multiply $\mathbf{f}_q^b$ with the particle volume. Quantities with function signatures (i.e., $\nabla p, \mathbf{f}^\sigma$ in Eq. 35, and $\phi^w, \mathbf{u}^w$ in Eqs. 36-37) need to be interpolated, which can be done using Eq. 34, except for the surface tension force which is outlined in §5.4.

We will leverage the same discrete per-particle drag force as Wretborn, Flynn, and Stomakhin [WFS22, Eq. 5]. They define

$$\mathbf{D}_q(\Delta\mathbf{u}) = \frac{\mu r_q \pi}{V_q}\left(6 + \frac{\rho^w r_q||\Delta\mathbf{u}||}{2\mu}\right)\Delta\mathbf{u} \tag{36}$$

for the local relative velocity $\Delta\mathbf{u} = \mathbf{u}^w(\mathbf{x}_q) - \mathbf{u}_q^b$. Since no drag forces on bubbles can arise in either pure air ($\phi^w = 0$) or pure water ($\phi^b = 0$) regions, we additionally scale the expression in Eq. 36 by (the unitless) water fraction, setting

$$\mathbf{d}_q^b \equiv \left(\frac{\mathbf{d}^b}{\phi^b}\right)\Bigg|_{\mathbf{x}_q} = \phi^w(\mathbf{x}_q)\mathbf{D}_q. \tag{37}$$

When interpolating $\phi^w(\mathbf{x}_q)$ using Eq. 34 the result is a vector (i.e., per-dimension water fractions), and when performing Eq. 37 discretely the multiplication should be interpreted as component-wise. The expression for the drag gradient is found using the chain rule as $\partial_{\mathbf{u}^b}\mathbf{d}_q^b \equiv (\partial_{\mathbf{u}^b}\mathbf{d}^b/\phi^b)|_q = \phi^w(\mathbf{x}_q)\partial_{\mathbf{u}^b}\mathbf{D}_q$ and

$$\partial_{\mathbf{u}^b}\mathbf{D}_q = -\frac{\mu r_q\pi}{V_q}\left(6 + \frac{\rho^w r_q||\Delta\mathbf{u}||}{2\mu}\right)\mathbb{I} - \left(\frac{\pi\rho^w r_q^2}{2V_q}\right)\frac{\Delta\mathbf{u}\otimes\Delta\mathbf{u}}{||\Delta\mathbf{u}||} \tag{38}$$

where $\otimes : (\mathbb{R}^3, \mathbb{R}^3) \to \mathbb{R}^{3\times 3}$ denotes the outer product.

5.3. Drag force rasterization

The preceding section established that the drag force on a particle q is $\mathbf{d}_q^b V_q = \phi^w(\mathbf{x}_q)\mathbf{D}_q V_q$. The ϕ^w multiplier avoids the singularity for purely bubble regions: there should not be any drag if there is no water present. However, when transferring the drag force from particle q to the grid $\mathcal{T}$ we can still encounter a scenario where a component of $\phi^w(\mathbf{x}_q) > 0$, but one of the affected faces τ has $\phi_{q\tau}^w = 0$. Naive weighted rasterization in this case could cause a nonzero rasterized force to be applied to a face with no water, which is clearly undesirable. Hence we propose an alternative method: we first rasterize $\mathbf{D}_q$ as

$$D_\tau = \frac{1}{V_\tau}\sum_q (\mathbf{D}_q\cdot\mathbf{e}_\tau)V_q\omega_{q\tau} \tag{39}$$

and afterwards multiply the result on the grid by ϕ_τ^w. Equation 39 guarantees that the total force received by the grid matches that on the particle; a proof is available in App. B. The effect is that we have modified the rasterization weights to avoid a potential singularity. The water drag force density contribution needed in Eq. 11 is then computed as $\mathbf{d}_\tau^w = -\phi_\tau^w D_\tau$.

We will treat the drag gradient in a similar manner, with the exception that we rasterize it lumped along rows as

$$\partial_{\mathbf{u}^b}\mathbf{D}_{\tau,\alpha\alpha} = \frac{1}{V_\tau}\sum_{q\in\mathcal{P}}\sum_{\beta=1}^{3}|\partial_{\mathbf{u}^b}\mathbf{D}_{q,\alpha\beta}V_q|(\mathbf{e}_\tau\cdot\mathbf{e}_\beta)\omega_{q\tau},\ \alpha\in[1,2,3], \tag{40}$$

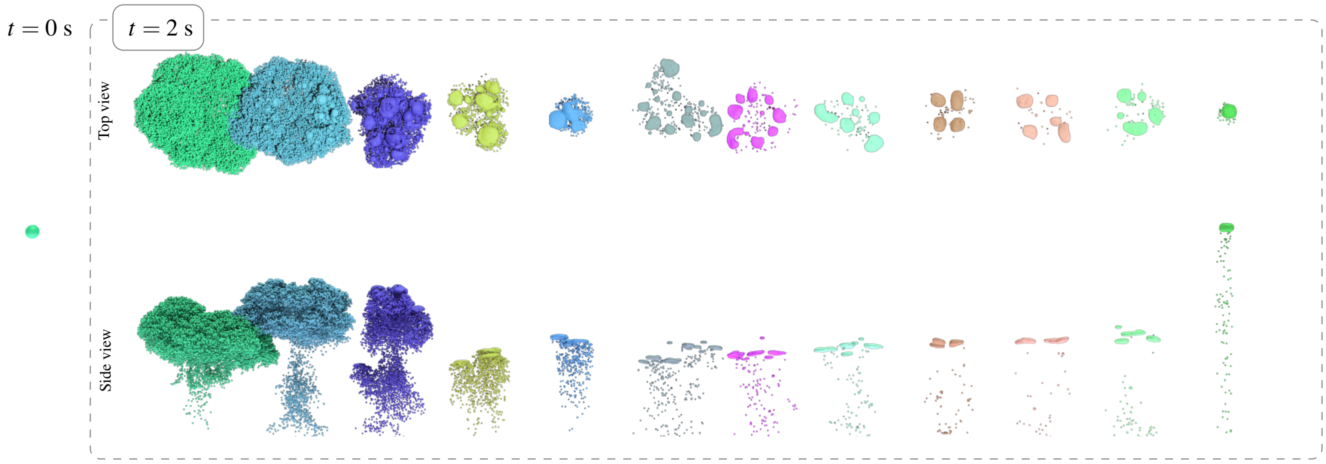

Figure 9: ***Surface tension.*** We study the effect of changing the surface tension coefficient. The leftmost simulation uses $\sigma = 0$, and each subsequent simulation increments the surface tension coefficient by 0.001 N/m. Particles are drawn with radius $2r_q$ for better visibility. ©Wētā FX

where α and β in Eq. 40 should be viewed as matrix indices, and we have slightly abused our notation to let $\mathbf{e}_\beta$ be the standard Euclidean basis vectors ($\mathbf{e}_1$ for x, $\mathbf{e}_2$ for y, etc). Then, the drag force density gradient on the grid is $\partial_{\mathbf{u}^w} \mathbf{d}_\tau^w = \phi_\tau^w \partial_{\mathbf{u}^w} \mathbf{D}_\tau = -\phi_\tau^w \partial_{\mathbf{u}^b} \mathbf{D}_\tau$.

This scheme ensures that $\partial_{\mathbf{u}^w} \mathbf{d}_\tau^w$ is a strictly negative diagonal matrix, so we can treat it as a vector, storing a single scalar value per face τ. The physical interpretation of Eq. 40 can be understood by considering Eq. 23. Since every component in $-\partial_{\mathbf{u}^w} \mathbf{d}_\tau^w$ is greater than zero, this results in a larger $\partial_{\mathbf{u}^w} \mathbf{f}_\tau^w$ (Eq. 12), and will cause a more conservative prediction of the water velocity $\bar{\mathbf{u}}_\tau^w$. This results in fewer oscillations from drag forces—which are stiff due to the nonlinear formulation—and a more stable system overall.

Fei, Batty, Grinspun, and Zheng [FBGZ18, §4.2.1] similarly propose to lump the drag force gradient, but do so for both phases and without taking the absolute value of the terms in Eq. 40. Empirically, we did not see large differences between the two implementations, but our version has the added benefit of avoiding a potential null mode when $\partial_{\mathbf{u}^w} \mathbf{d}^w = \frac{\phi^w \rho^w}{\Delta t} \mathbb{I}$.

5.4. Surface tension force

When discretizing Eq. 5 we have the choice to leverage the ϕ_τ^b defined on the grid or compute the force from the particle geometry directly. The latter option would result in a formulation similar to SPH surface tension models that use a color field (e.g. [MCG03]). We adopt the former option, since surface tension and pressure forces are intimately linked and we leverage the grid to compute the pressure.

The fractions ϕ_τ^b are placed irregularly in space since they are stored on faces. To compute curvature we need first- and second-order derivatives defined at the same location; and to avoid a costly re-rasterization of particle volume to cell centers, we average the fractions to the cell centers by

$$\phi_{\mathbf{i}}^b = \sum_{\tau \in \mathcal{N}_{\mathbf{i}}} \frac{\psi_\tau \phi_\tau^b}{6}, \tag{41}$$

where $\psi_\tau \in \{0, 1\}$ is a filter defined per face, which we will use to remove single-particle regions on the grid, and $\mathcal{N}_{\mathbf{i}}$ is the set of faces belonging to cell $\mathbf{i}$. The filter is fully defined later in §5.4.1.

We then use the standard cell-centered central finite difference stencil and define the averaged fraction gradient $\nabla \phi_{\mathbf{i}}^b$ and the corresponding Hessian $\mathbb{H}_{\mathbf{i}}$. The mean curvature follows from expanding the expression in Eq. 5 using the chain rule as

$$\kappa_{\mathbf{i}} = \frac{1}{||\nabla \phi_{\mathbf{i}}^b||} \left(\operatorname{trace}(\mathbb{H}_{\mathbf{i}}) - \frac{(\nabla \phi_{\mathbf{i}}^b)^T \, \mathbb{H}_{\mathbf{i}} \, \nabla \phi_{\mathbf{i}}^b}{||\nabla \phi_{\mathbf{i}}^b||^2} \right).$$

Finally we compute the surface tension force density per face $\tau = (i - \frac{1}{2}, j, k) \in \mathcal{T}_x$ leveraging the neighboring cell values of curvature and bubble fraction as

$$f_\tau^\sigma = \sigma \left(\frac{\kappa_{(i,j,k)} + \kappa_{(i-1,j,k)}}{2} \right) \left(\frac{\phi_{(i,j,k)}^b - \phi_{(i-1,j,k)}^b}{h} \right), \tag{42}$$

and similarly for the other dimensions. The per-phase surface tension force densities can now be computed as $\phi_\tau^b f_\tau^\sigma$ and $\phi_\tau^w f_\tau^\sigma$.

Equation 42 relies on two operations that have a smoothing effect. The first is the construction of the cell-centered fraction $\phi_{\mathbf{i}}^b$ in Eq. 41, and the second is the averaging of the cell-centered curvature $\kappa_{\mathbf{i}}$ directly to faces in Eq. 42. We explored avoiding this by computing curvature directly on the faces, and did so by defining finite difference stencils centered at each face, but we observed no benefits from this approach. The resulting curvature was noisy, leading to spurious movement of some particles and requiring additional filtering to be practical. As a result, we abandoned this approach.

5.4.1. Numerical considerations

Surface tension as a phenomenon is momentum conserving when considering any connected bubble region. However, given our explicit discretization of surface tension as a force, small errors inevitably arise when integrating the force numerically. We propose two practical solutions: a bubble fraction filtering algorithm, which removes the grid footprint of isolated single-cell regions, and a drift compensation strategy.

Fraction filtering. We have found that a single particle, or many particles confined to an isolated voxel, can cause spurious errors when numerically integrating the surface tension force. These errors manifest as rising bubbles getting stuck or moving erratically, as a result of the diffuse bubble representation on the grid being under-resolved. We propose to solve it using the following filtering algorithm.

To make our notation concise, let $\mathbf{a},\mathbf{b},\mathbf{c} \in \mathcal{C}$ be indices representing cells. We will overload the notation somewhat and let $\tau(\mathbf{a},\mathbf{b}) \in \mathcal{T}$ represent the face neighboring both cells $\mathbf{a}$ and $\mathbf{b}$. Then we define ψ_τ using the following procedure:

1. **Mark cells.** If a cell $\mathbf{a}$ has $\phi^b_\tau > 0$ for all neighboring faces $\tau \in \mathcal{N}_\mathbf{a}$ mark it with a unique identifier $\mathrm{id}_\mathbf{a} \in \mathbb{N}$. Otherwise set $\mathrm{id}_\mathbf{a} = -1$.
2. **Unmark connected cells.** If two marked cells $\mathbf{a},\mathbf{b}$ share a face, set $\mathrm{id}_\mathbf{a} = \mathrm{id}_\mathbf{b} = -1$.
3. **Dilate.** If an unmarked cell $\mathbf{a}$ neighbors only one marked cell $\mathbf{b}$, and there are at at least 2 neighboring faces $\tau \in \mathcal{N}_\mathbf{a}$ with $\phi_\tau > 0$; set $\mathrm{id}_\mathbf{a} = \mathrm{id}_\mathbf{b}$.
4. **Dilate.** If an unmarked cell $\mathbf{a}$ neighbors two marked cells $\mathbf{b},\mathbf{c}$ such that $\phi^b_{\tau(\mathbf{a},\mathbf{b})} > 0$ and $\phi^b_{\tau(\mathbf{a},\mathbf{c})} > 0$, and $\mathrm{id}_\mathbf{b} = \mathrm{id}_\mathbf{c}$; then set $\mathrm{id}_\mathbf{a} = \mathrm{id}_\mathbf{b}$.
5. **Filter.** For every face $\tau(\mathbf{a},\mathbf{b})$: if $\mathbf{a}$ or $\mathbf{b}$ is marked, set $\psi_\tau = 0$; otherwise $\psi_\tau = 1$.

The proposed algorithm was developed by inspecting the footprint of a single particle when rasterized to the grid. It finds any isolated voxel that contains particles, and then performs a topological dilation on the grid. A schematic image of the filtering algorithm is presented in Figure 11. Since the dilation operations extend independently in each dimension, two dilation operations are enough to touch all rasterized faces for a single particle even in three dimensions.

Drift compensation. Let $\mathcal{O}_\nu \subset \mathcal{T}$ be the largest possible set of faces such that they all have nonzero bubble fraction $\phi^b_\tau > 0, \forall\tau \in \mathcal{O}_\nu$ and any two faces in $\mathcal{O}_\nu$ can be traversed via cells that neighbor $\mathcal{O}_\nu$. That is, two faces $\tau_1,\tau_m \in \mathcal{T}$ are connected if there exists a path of cells $(\mathbf{i}_1,\ldots,\mathbf{i}_m) \in \mathcal{C}$ such that the shared face $\phi^b_{\tau(\mathbf{i}_\alpha,\mathbf{i}_{\alpha+1})} > 0\ \forall\alpha \in [1,m-1]$ and $\tau_1 \in \mathcal{N}_{\mathbf{i}_1}, \tau_m \in \mathcal{N}_{\mathbf{i}_m}$. The set $\mathcal{O}_\nu$ can be created by a standard flood-fill algorithm where faces with nonzero bubble fraction are nodes and cells are edges. We call ν a "pocket", and only consider pockets with more than one face, $|\mathcal{O}_\nu| > 1$.

For the surface tension force to be momentum conserving we require that it integrates to zero for all pockets ν. Discretely, this condition may not be satisfied, and there might be some nonzero force residual $\mathbf{R}_\nu$ per pocket

$$\mathbf{R}_\nu = \sum_{\tau\in\mathcal{O}_\nu} f^\sigma_\tau V_\tau. \tag{43}$$

By introducing the force density

$$\chi_\nu = \frac{\mathbf{R}_\nu}{\sum_{\tau\in\mathcal{O}_\nu} V_\tau} \tag{44}$$

we can guarantee that the discrete shifted surface tension force integral is zero, since

$$\sum_{\tau\in\mathcal{O}_\nu} V_\tau \left(f^\sigma_\tau - \chi_\nu\right) = \mathbf{R}_\nu - \mathbf{R}_\nu = \mathbf{0} \tag{45}$$

for any pocket ν. The resulting particle corrected surface tension force per unit of bubble volume takes the form

$$\mathbf{f}^\sigma(\mathbf{x}_q) = \sum_{\tau\in\mathcal{T}} \phi^c_\tau (f^\sigma_\tau - \sum_{\nu|\tau\in\mathcal{O}_\nu} \chi_\nu)\omega_{q\tau}, \tag{46}$$

where we note that every particle necessarily will only rasterize to a single pocket; thus the set $\{\nu \mid \tau \in \mathcal{O}_\nu\}$ will always contain a single entry when $\omega_{q\tau} > 0$. Note also that the interpolation in Eq. 46 is clamped, as otherwise the total surface tension force computed on the particles would be overestimated. We show in App. C that applying Eq. 44 ensures momentum conservation even when integrating over the different discrete representations.

5.5. Pore-pressure projection

The discrete systems of equations in Eqs. 24-26 can be solved by slightly modifying a standard staggered pressure projection solver.

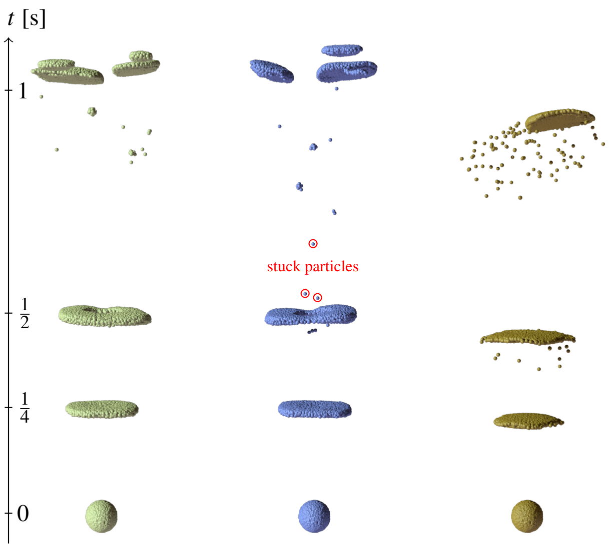

Figure 10: ***Surface tension ablation.*** We show the efficacy of our numerical techniques to improve the integration of surface tension forces. Each column contains four image snapshots at different times of the same simulation, taken from a fixed camera. Left: our method as presented in Alg. 1. Middle: skip the *Fraction filtering* by setting $\psi = 1$ in Eq. 41. The circled in red particles are "stuck" (have zero velocity), and are part of the $t = 1$ snapshot. Right: skip the re-application of surface tension forces in Eq. 51, and instead rasterize pre-projection bubble velocity $\bar{\mathbf{u}}^b$ (Eq. 22). ©Wētā FX

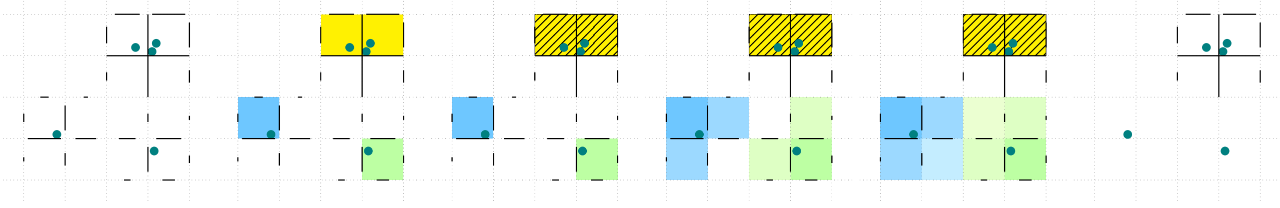

Figure 11: ***Filtering illustration.*** Bubble particles are marked as circles, and the rasterized fractions ϕ^b_τ are visualized as lines on each grid face. Each unique identifier is represented by a different color (blue, green, yellow). To highlight the different dilation operations in steps (3) and (4), each subsequent step receives a lighter hue. The unmarking of cells in step (2) is represented by the striped diagonal pattern. ©Wētā FX

Let $P \in \mathbb{R}^{|\mathcal{C}|}$ be a column-vector of all unknown pressure values and $\mathbf{U}^w, \mathbf{U}^b \in \mathbb{R}^{|\mathcal{T}|}$ column-vectors of all velocity values. We will follow the same notation for the different stages of velocity by means of (time) subscripts or bar/hat modifiers. For example, $\mathbf{U}^w_{n+1}$ are the staggered water velocity values at the end of the time step, and $\bar{\mathbf{U}}^b$ are the bubble velocity values fed into the pressure projection algorithm. Furthermore, we let $\mathbf{\Phi}^b, \mathbf{\Phi}^w \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{T}|}$ be the diagonal bubble and water fraction matrices, respectively, and similarly for masses $\mathbf{B}, \mathbf{W} \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{T}|}$. Finally, we use the standard staggered finite difference discretizations of the gradient $\mathbf{G} : \mathcal{C} \to \mathcal{T}$ and divergence $\mathbf{G}^T : \mathcal{T} \to \mathcal{C}$ [HW65].

Then, the discrete equivalent of Eqs. 24-26 is

$$\begin{aligned}
\mathbf{\Phi}^b\mathbf{B}\left(\hat{\mathbf{U}}^b - \bar{\mathbf{U}}^b\right) &= -\Delta t \mathbf{\Phi}^b \mathbf{G} P_{n+1}, \\
\mathbf{\Phi}^w\mathbf{W}\left(\mathbf{U}^w_{n+1} - \bar{\mathbf{U}}^w\right) &= -\Delta t \mathbf{\Phi}^w \mathbf{G} P_{n+1}, \\
\mathbf{0} &= \mathbf{G}^T\left(\mathbf{\Phi}^b\hat{\mathbf{U}}^b + \mathbf{\Phi}^w\mathbf{U}^w_{n+1}\right).
\end{aligned}$$

Substitution of the first two rows into the third and reordering yields

$$\mathbf{G}^T\left(\mathbf{\Phi}^b\mathbf{B}^{-1} + \mathbf{\Phi}^w\mathbf{W}^{-1}\right)\mathbf{G}P_{n+1} = \Delta t \mathbf{G}\left(\mathbf{\Phi}^b\bar{\mathbf{U}}^b + \mathbf{\Phi}^w\bar{\mathbf{U}}^w\right). \quad (47)$$

By setting $\mathbf{M}^{-1} = \mathbf{\Phi}^b\mathbf{B}^{-1} + \mathbf{\Phi}^w\mathbf{W}^{-1}$ we see that the previous system of equations is a standard variable density Poisson problem, and can be solved efficiently by e.g., a preconditioned CG iterative solver. As a result, extending a standard single-phase pressure solver to this two-phase pore-pressure method requires little work: one must include degrees of freedom for both the air and water phases, rather than water alone, and modify the mass and divergence computations according to Eq. 47.

The final water velocity is given by

$$\mathbf{U}^w_{n+1} = \bar{\mathbf{U}}^w - \Delta t \mathbf{W}^{-1}\mathbf{G}P_{n+1} \quad (48)$$

which yields the individual face values $u^w_{n+1,\tau}$. We reiterate that in the inertia-aware schemes the intermediate particle velocities $\hat{\mathbf{U}}^b$ are discarded after the solve, and we can simply ignore the pressure gradient subtraction for the bubble phase.

Hydrostatic boundary condition. We follow Stomakhin, Wretborn, Blom, and Daviet [SWBD20] and use Dirichlet hydrostatic pressure boundary conditions around the exterior borders of the sparsely allocated volumetric tiles. For any cell $\mathbf{i} \in \mathcal{C}$ that neighbors a non-allocated cell we set

$$p_{\mathbf{i}} = \rho^w(\mathbf{g}\cdot\mathbf{x}_{\mathbf{i}}). \quad (49)$$

To ensure that the mixture density is exactly ρ^w by the boundary we allocate sufficient tiles such that no particle would rasterize outside of a tile or to a cell on the border of an existing tile. For example, the padding in Fig. 12 is zero which is generally too small; we typically use a padding of 1-4 tiles. As the padding becomes larger the simulation becomes slower, but our choice of a hydrostatic boundary condition becomes more correct. Empirically, we have found that the hydrostatic *pressure* condition allows us to use fewer tiles than

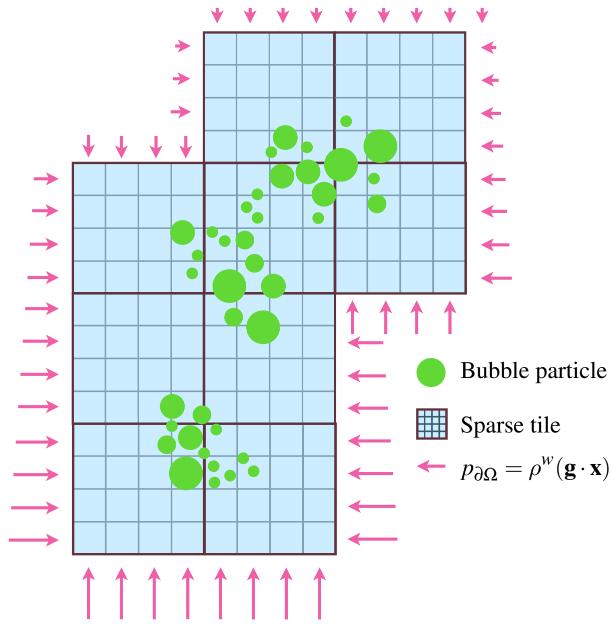

Figure 12: ***Sparse layout.*** Sparse volumetric tiles are created around particles such that no particle rasterizes to faces outside of the simulation region. The arrows around the boundary represent the (scalar) hydrostatic pressure boundary condition Eq. 49. ©Wētā FX

a zero *velocity* boundary condition. Since the pressure boundary condition allows for velocity flux across the boundary, the buoyancy effects are less inhibited. The result is that bubbles appear to rise more energetically.

We have additionally found that using the hydrostatic pressure condition as the initial guess for pressure improves the stability of the solve. That is, we set the pressure value $p_{0,\mathbf{i}}$ at the beginning of each Newton solve for each cell $\mathbf{i} \in \mathcal{C}$ using Eq. 49.

Discrete equilibrium between surface tension and pressure. As argued in §3.1 a static equilibrium for pressure and surface tension forces exists only if $\nabla p = \mathbf{f}^{\sigma}$ for all $\mathbf{x} \in \Omega$, the discrete version of which is

$$\nabla p_\tau = f_\tau^{\sigma} \quad \tau \in \mathcal{T}. \tag{50}$$

Equation 50 can only be satisfied if the pre-projection velocities $\bar{\mathbf{u}}_\tau^b$ and $\bar{\mathbf{u}}_\tau^w$ receive the impulse $(\Delta t/\rho^\star) f_\tau^\sigma$ for the bubble and water phases, respectively. This means it is not possible to directly rasterize the pre-projection particle velocity $\bar{\mathbf{u}}^b$ as suggested in Eq. 22, since the interpolation of Eq. 46 and subsequent rasterization by Eq. 33 will not preserve the surface tension contribution exactly on the grid. Instead, we subtract the integrated surface tension force on the particles and re-apply it to the grid. Using the notation introduced in Eq. 22 and Eq. 33 we have

$$\bar{u}_\tau^b = \underbrace{\left(\mathbf{u}_{n+1,q}^b + \frac{\Delta t}{\rho_q^b}\nabla p_n(\mathbf{x}_q) - \frac{\Delta t}{\rho_q^b}\mathbf{f}^{\sigma}(\mathbf{x}_q)\right)_\tau^b}_{\text{particles to grid transfer}} + \frac{\Delta t}{\rho_\tau^b} f_\tau^\sigma. \tag{51}$$

Equation 51 ensures that, from the perspective of the pressure solver, the applied surface tension force for the two phases is consistent.

5.6. Advection

We will advect the two phases separately at the start of the time step. For bubbles we directly advect particles using RK1, i.e., $\mathbf{x}_q \leftarrow \mathbf{x}_q + \Delta t \mathbf{u}_q^b$. We use MacCormack advection [SFK*08] to advect the water velocity.

Algorithm 1 Modified Newton solver

Advect phases separately. (§5.6)
Rasterize $\phi_\tau^b, \phi_\tau^w, \rho_\tau^b$. (Eq. 30, Eq. 31, Eq. 32)
Initialize pressure $p_{n=0,\mathbf{i}}$. (Eq. 49)
Compute surface tension force f_τ^σ. (Eq. 42 using Eq. 41)
for Newton iteration $n = 0, 1, \ldots$ **do**
  [a] Compute drag force density $\mathbf{d}_{n,q}^b$ and gradient $\partial_{\mathbf{u}^b}\mathbf{d}_{n,q}^b$. (Eqs. 36-38)
  [b] Compute total force density $\mathbf{f}_{n,q}^b$ and gradient $\partial_{\mathbf{u}^b}\mathbf{f}_{n,q}^b$. (Eq. 35 using Eq. 46)
  [c] Solve for $\mathbf{u}_{n+1,q}^b$. (Eq. 21)
  [d] Rasterize pre-projection velocity $\bar{u}_\tau^b$. (Eq. 51)
  [e] Rasterize drag D_τ^b and $\partial_{\mathbf{u}^b} D_\tau^b$. (Eq. 39, Eq. 40)
  [f] Compute total force density $f_\tau^w(u_{n,\tau}^w, 0)$ and gradient $\partial_{\mathbf{u}^w} f_\tau^w(u_{n,\tau}^w)$. (Eq. 11)
  [g] Compute pre-projection velocity $\bar{u}_\tau^w$. (Eq. 23)
  [h] Solve for $u_{n+1,\tau}^w$. (Eq. 47, Eq. 48)
end for

6. Implementation

We implemented the full algorithm as presented in Fig. 5 and Alg. 1 by leveraging a preexisting FEM solver [LSD*22] for (5c) as well as a pressure solver with support for variable density fluids [SLW*23] for (5h), but any equivalent tools would suffice. The additional developments needed are the force computations (4, 5a, 5b, 5f) as well as the rasterization routines (2, 5d, 5e).

We render bubbles by rasterizing particles to a distance field, often to a finer grid than the simulation grid. Since the rasterization operation can be costly we split particles into two groups: *isolated* or *connected*. Using a simulation grid resolution h, the groups are determined by creating an initial level set as the union of SDF spheres with radius $r_h^{\text{surfacing}} = \sqrt{3/4}h$ centered at each particle. The level set is eroded by $r_h^{\text{surfacing}}$, renormalized, and then dilated by $r_h^{\text{surfacing}}$. Finally we sample the level set at each particle position; if the particle falls outside of the zero iso-contour it is considered an isolated bubble, and otherwise part of a connected bubble.

Isolated particles are represented as spheres. *Connected* particles are represented by a level set which we construct as the union of oriented ellipsoids [YT13; SLW*23]. We often benefited visually from rasterizing to a finer grid than the simulation grid size; in Fig. 3 we used $h/2$ and in Fig. 8 we used $h/3$. We used the basic liquid shader in SideFX Houdini [Hou] for both particles and the level set, and rendered everything in Mantra.

7. Results and discussion

We demonstrate the applicability of our method by reproducing the "hero" bubble examples from Stomakhin, Wretborn, Blom, and Daviet [SWBD20]: an overturning barrel of air (Fig. 1, Fig. 3, Fig. 15) and a person exhaling under water (Fig. 8). Our method can also capture diffuse effects, such as the sand-air-water mixture in Fig. 2 or the bubble column in Fig. 13a. Simulation time is typically dominated by the pressure solver and the need to perform at least two pressure projections per time step, due to the Newton loop. Table 2 lists performance numbers and settings used. All simulations were run on a 32-core machine.

Surface tension. To demonstrate the effect of surface tension in our method we provide two parameter studies in Fig. 9 and Fig. 7. Our method qualitatively reproduces many bubble shapes found in nature, and the fact that axisymmetric bubbles rise faster than asymmetric bubbles [TSG15, see Fig. 1 and Fig. 10].

Our chosen surface tension model is known to be more diffusive than models with a sharp interface [HSKF07]. Figure 4 shows a two-phase simulation with equal densities from an initial perfect cube at rest. The leftmost example uses a hybrid marker-based method with a sharp interface surface tension model that has been verified against theoretical predictions [BB12, §4]. The image is taken at 0.25 s, after which the three versions begin to diverge. In particular, our results are noticeably noisier.

CSF models are also known to suffer from parasitic currents from the curvature estimate ([Pop09; KLK20]) and the difficulty of exactly counterbalancing pressure forces. Common mitigation strategies rely on reconstructing a sharp interface from the vol-

Table 2: Performance numbers and settings used for our examples. The last column shows the average time breakdown for different functions: P - pressure projection, PP - power diagram generation, I - interpolation/rasterization, C - collision resolution. We do not label functions if they are below 10% of the total time. Clarifications: [1] Denoting basis by (T)rilinear, (P)ower. [2] The number of time subdivisions per frame. [3] The number of Newton iterations per substep.

Example	ρ [kg/m^3]	σ [N/m]	h [cm]	r_q [cm]	Λ_q[1]	FPS	Sub.[2]	Nwt.[3]	[s] / frame	[%]
Fig. 1: Overturning barrel	1 : 10	0.5	1	$r_h^{\text{surfacing}}$	T	48	30	2	1312	P90
Fig. 2: Bubble-sand mixture	1 : 1000 : 1500	0	1	$[0.05, 0.5]$	T	48	2	2	17	P50/C30
Fig. 3: Barrel timelapse	1 : 10	0.072	1	$r_h^{\text{surfacing}}$	T	96	10	2	601	P90
Fig. 7: Bubble sizes	1 : 10	0.0072	0.125	$r_h^{\text{surfacing}}$	T	48	30	3	$[6, 949]$	P90
Fig. 8: Exhale	100 : 1000	0.36	0.1	$\sqrt[3]{\frac{3(h/2)^3}{4\pi}}$	P	48	20	2	798	P15/PP30/I45
Fig. 9: Surface tension	1 : 10	$[0, 0.011]$	0.125	$r_h^{\text{surfacing}}$	T	38	30	3	$[68, 671]$	P90
Fig. 13a: Diffuse bubble column	1 : 1000	0	2	$[0.05, 0.5]$	T	24	2	2	0.1	–
Fig. 14: Effect of particle size	100 : 1000	–	0.1	–	T,P	24	20	2	$[7, 90]$	–

ume fraction field, which is not possible in our mixture context due to the absence of a well-defined interface. In practice, the discrete equilibrium discussed in §5.5 is difficult to achieve. Popinet [Pop09, §4] points out that the discrete equilibrium is achievable for continuum surface force models when (1) the discrete gradient operator in Eq. 50 for pressure and surface tension force are identical and (2) the curvature is precisely constant. Since ϕ^b and p are not stored at the same discrete locations (faces and cells, respectively) we rely on the averaging in Eq. 41, which results in a blurred finite-difference stencil, thereby violating condition (1). We have investigated a cell-centered rasterization scheme for ϕ^b for which a discrete equilibrium *can* be found, but at this point condition (2) becomes the limiting factor. In particular, the curvature computation used in Eq. 42 does not provide a sufficiently accurate curvature metric for under-resolved fractional fields. Such configurations are common, since single bubbles often tear away from larger air regions and rise individually, causing a topology not dissimilar from the schematic in Fig. 11. We found that the same filtering and drift compensation strategies outlined in §5.4.1 were needed; we therefore opted for our staggered rasterization technique, since it improves the accuracy of the pressure solver. Ultimately, the discrete equilibrium is not achievable for our scheme in practice, but since the air-water dynamics we are interested in are turbulent in nature, we have not found this to be a significant limitation.

We show the efficacy of our surface tension force treatment by the ablation study in Fig. 10. Without the proposed filtering in Eq. 41, stray isolated bubble particles tend to become stuck with zero velocity (Fig. 10, middle). Our proposed bubble velocity rasterization in Eq. 51, which ensures that the surface tension force impulse on the grid is consistent for the two phases, results in more cohesive bubble shapes (Fig. 10, right). The *Drift compensation* technique we propose (Eq. 46) was not visually apparent for the same initial configuration as the other examples in Fig. 10, so we omitted it from this comparison; however, the technique was important in equal-density examples like Fig. 4 to prevent the bubble's center of mass from drifting.

Comparison to [WFS22]. In a purely diffuse scenario ($\phi^b \ll 1$) where the fluid motion is calm, our method produces results (Fig. 13a) similar to that of Wretborn, Flynn, and Stomakhin [WFS22] (Fig. 13b). However, their method can exhibit unnatural sinking due to the correction of the water mass matrix in Eq. 19, which effectively increases the local density of water with the lumped drag force matrix (Eq. 40). This artifact can appear in turbulent scenarios where drag forces are large, and are also exacerbated by large bubble densities as exemplified in Fig. 13d; although $\rho^b < \rho^w$ the bubbles sink due to the non-zero drag force gradient in Eq. 38. Our method overcomes this issue, as demonstrated in Fig. 13c, by implicitly integrating drag forces in Eq.25.

Comparison to marker-particle methods. Compared to standard marker-particle hybrid fluid solvers, which commonly default to 8 particles per voxel and a particle radius tied to the grid resolution ([ZB05; JSS*15; FHA17]), the hybrid method we present has no such restriction. We have, however, found that the choice of particle size does affect the kind of dynamics that are possible. Consider Fig. 14. We release 5 cm^3 of still air in the shape of a sphere. The grid resolution is $h = 1$ mm, and we sample particles $h/2$ apart (i.e., there are 8 particles per voxel). Arguably, the most natural choice is to set the volume of each particle to that of the subdivision, $V_q = h^3/8$; the two rightmost columns in Fig. 14 show this case. When the surface tension is zero (top row), it is clear that there is a lack of

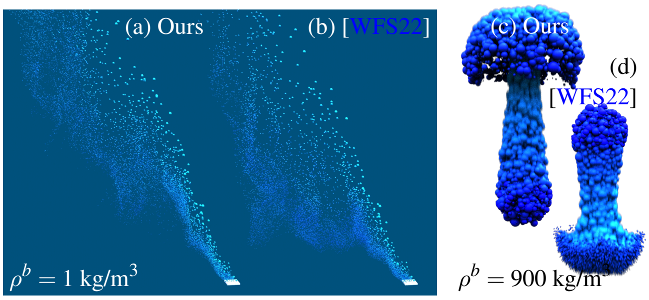

Figure 13: ***Diffuse bubble comparison.*** We compare our method to [WFS22]. Our method qualitatively produce the same diffuse effects (Fig. 13a-13b), but avoids sinking artifacts (Fig. 13c-13d). Water density is $\rho^w = 1000$ kg/m^3 in all examples. Particles are drawn with radius $2r_q$ and colored according to their speed. ©Wētā FX

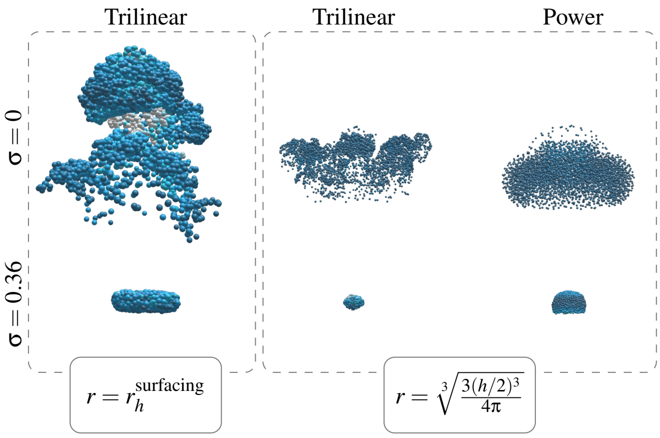

Figure 14: ***Effect of particle size.*** A result of particles being massful is that particle radii are decoupled from the grid size and can be chosen freely. Presented are six simulations with $h = 1$ mm taken at $t = 0.5$ s. Particles are initialized by sampling a sphere with radius 1.06 cm at $h/2$ intervals. ©Wētā FX

"hero" dynamics: particles quickly spread out and do not interact much.

In marker-particle schemes the effective particle radii are much larger than described above. In order to not disappear on the grid, the minimum radius of a particle is the distance from the voxel center to the corner, $r_h^{\text{surfacing}} = \sqrt{3/4}h$. Using the $r_h^{\text{surfacing}}$ particle radius with our method, connected bubble shapes naturally form even in the absence of surface tension (Fig. 14, top-left). In this configuration, as individual particles eject from the group they still represent a (comparatively) large region of air. The lower density forms an air channel, allowing nearby particles to flow along the channel more easily than if they were penetrating further into the water. For this reason, whenever we use the *Trilinear* method and we seek surface tension effects, we tend to use particle radii $r_h^{\text{surfacing}}$.

Note that, like in marker schemes, using $r_h^{\text{surfacing}}$ in our method "over-prescribes" the amount of air in the simulation by a factor

$$\frac{4\pi\left(r_h^{\text{surfacing}}\right)^3/3}{h^3/8} \approx 22,$$

and is made possible by our introduction of the bubble fraction clamp ϕ^c. Effectively, a massful particle that is "too large" behaves like a marker particle.

We provide a qualitative comparison between our method and that of Stomakhin, Wretborn, Blom, and Daviet [SWBD20], which leverages a marker-particle-based discretization, in Fig. 15. The geometric shapes produced by the two methods are similar, although the speed with which the column of air rises is slower with our method; we suspect the difference is due to the addition of the drag force model in Eq. 7. This highlights one traditional challenge of combining distinct diffuse and hero models: augmenting the method of Stomakhin, Wretborn, Blom, and Daviet [SWBD20] with additional diffuse bubbles would prove challenging, since the sub-grid dynamics model for tiny particles may cause them to rise at rates inconsistent with their grid-scale counterparts. Our model, since it relies on a unified underlying physical model and discretization for both diffuse and hero regimes, does not have the same difficulty.

The *Power* method does not leverage the fraction clamp, and thus the real volume (i.e., $h^3/8$ for 8 particles per voxel) must be used. Although it is diffusive, similar to the *Trilinear* method with "small" particles, the *Power* method tends to behave better with the surface tension force: the motion is typically less jittery, and the formed bubble shapes are smoother. We attribute this to the regularization strategy of moving particle positions to the center of power cells, which ensures that the bubble volume on the grid is conserved (up to numerical precision). A realistic scenario where we used the *Power* method is shown in Fig. 8.

We make a final point here that for any simulation where the fraction clamp $\phi^c \neq 1$, the bubble mass *on the grid* is not conserved due to the choice of dividing by unclamped fraction $\bar{\phi}$ in Eq. 32. It would be interesting to explore strategies that take the compressible nature of these configurations into account, but that would remove the "marker $\leftrightarrow$ massful" duality which has proven useful. We leave this for future work.

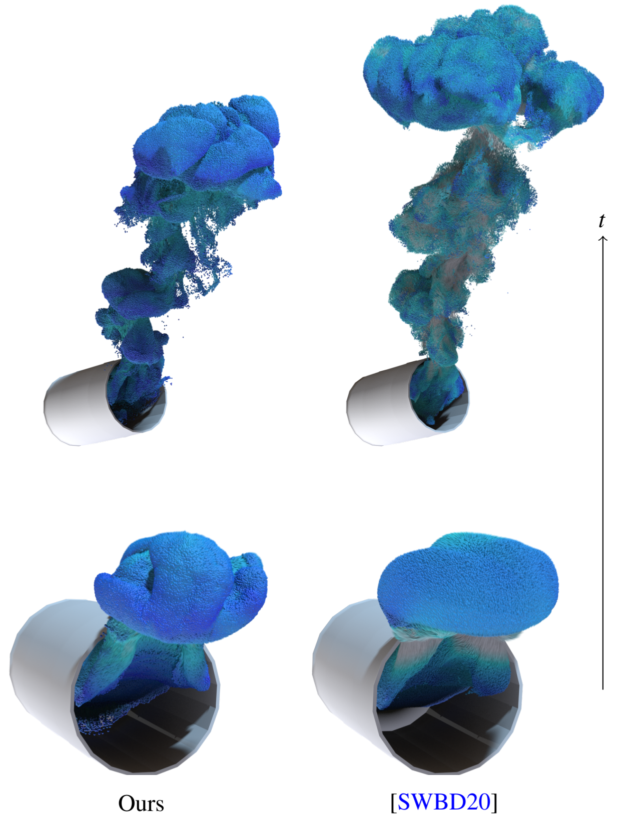

Figure 15: ***Barrel comparison.*** We recreate the overturning barrel from Fig. 3 using [SWBD20] (right column) and compare with our method (left column) at two different times (top and bottom, respectively). Particles are rendered with motion blur, and colored according to their speed. ©Wētā FX

8. Conclusion

We have proposed a new discretization for bubble dynamics using a hybrid mixture method where air is represented by massful particles surrounded by Eulerian water. The phases are coupled together in a modified *stabilized inertia-aware* Newton iterative scheme that interleaves particle force computations with an Eulerian variable density pressure projection. We additionally extended the continuum surface force model of surface tension [BKZ92] for our mixture context. Taken together, these contributions enable our method's key distinguishing property: the ability to smoothly reproduce the behavior of both sub-grid diffuse bubble and grid-resolved hero bubbles in a single unified discretization.

Our approach could benefit from a particle splitting/merging technique to keep particles at an optimal radius near hero regions and smaller in diffuse regions. This could potentially be achieved in a manner similar to transition models used for water-in-air states ([LSD*22; SLW*23; LTKF08]) or inter-particle interaction models [JS17].

Acknowledgments

The authors extend their gratitude to the Wētā FX leadership and Simulation department for supporting this work. This research was supported in part by grants from NSERC (RGPIN-2021-02524) and CFI-JELF (Grant 40132).

References

[AAT13] Akinci, NADIR, Akinci, GIZEM, and Teschner, MATTHIAS. "Versatile surface tension and adhesion for SPH fluids". *ACM Trans. Graph.* 32.6 (Nov. 2013), 1–8 3.

[AO96] Andrews, M J and O'Rourke, P J. "The multiphase particle-in-cell (MP-PIC) method for dense particulate flows". *Int. J. Multiphase Flow* 22.2 (Apr. 1996), 379–402 4.

[BB12] Boyd, LANDON and Bridson, ROBERT. "MultiFLIP for energetic two-phase fluid simulation". *ACM Trans. Graph.* 31.2 (Apr. 2012), 1–12 2, 3, 5, 13.

[BBB10] Brochu, TYSON, Batty, CHRISTOPHER, and Bridson, ROBERT. "Matching fluid simulation elements to surface geometry and topology". *ACM SIGGRAPH 2010 papers*. SIGGRAPH '10 Article 47. New York, NY, USA: Association for Computing Machinery, July 2010, 1–9 3.

[Bif] Bifrost. *Bifrost*. https://www.autodesk.com/products/maya/bifrost. Accessed: 2024-9-27 3.

[BK*04] Bardenhagen, SCOTT G, Kober, EDWARD M, et al. "The generalized interpolation material point method". *Computer Modeling in Engineering and Sciences* 5.6 (2004), 477–496 2, 7.

[BKZ92] Brackbill, J U, Kothe, D B, and Zemach, C. "A continuum method for modeling surface tension". *J. Comput. Phys.* 100.2 (June 1992), 335–354 2–5, 16.

[Dav20] Daviet, GILLES. "Simple and scalable frictional contacts for thin nodal objects". en. *ACM Trans. Graph.* 39.4 (Aug. 2020) 2.

[DB17] Daviet, GILLES and Bertails-Descoubes, FLORENCE. "Simulation of Drucker–Prager granular flows inside Newtonian fluids". en. (Feb. 2017) 2, 4.

[DBWG15] Da, FANG, Batty, CHRISTOPHER, Wojtan, CHRIS, and Grinspun, EITAN. "Double bubbles sans toil and trouble". en. *ACM Trans. Graph.* 34.4 (July 2015), 1–9 3.

[dGWH*15] De Goes, FERNANDO, Wallez, CORENTIN, Huang, JIN, et al. "Power particles: an incompressible fluid solver based on power diagrams". *ACM Trans. Graph.* 34.4 (July 2015), 1–11 3.

[DHB*16] Da, FANG, Hahn, DAVID, Batty, CHRISTOPHER, et al. "Surface-only liquids". en. *ACM Trans. Graph.* 35.4 (July 2016), 1–12 3.

[DKP16] Deike, LUC, Kendall Melville, W, and Popinet, STÉPHANE. "Air entrainment and bubble statistics in breaking waves". *J. Fluid Mech.* 801 (Aug. 2016), 91–129 2.

[EFFM02] Enright, DOUGLAS, Fedkiw, RONALD, Ferziger, JOEL, and Mitchell, IAN. "A Hybrid Particle Level Set Method for Improved Interface Capturing". *J. Comput. Phys.* 183.1 (Nov. 2002), 83–116 2, 3.

[FBGZ18] Fei, YUN (RAYMOND), Batty, CHRISTOPHER, Grinspun, EITAN, and Zheng, CHANGXI. "A multi-scale model for simulating liquid-fabric interactions". *ACM Trans. Graph.* 37.4 (July 2018), 1–16 4, 7, 10.

[FHA17] Fu, LIN, Hu, XIANGYU Y, and Adams, NIKOLAUS A. "A physics-motivated Centroidal Voronoi Particle domain decomposition method". *J. Comput. Phys.* 335 (Apr. 2017), 718–735 14.

[FSWW24] Flynn, SEAN, Stomakhin, ALEXEY, Wretborn, JOEL, and White, DANIEL. "Art Directable Underwater Explosion Simulation". *ACM SIGGRAPH 2024 Talks*. SIGGRAPH '24 Article 55. New York, NY, USA: Association for Computing Machinery, July 2024, 1–2 2, 3.

[GAB20] Goldade, RYAN, Aanjaneya, MRIDUL, and Batty, CHRISTOPHER. "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 2.

[GTJS17] Gao, MING, Tampubolon, ANDRE PRADHANA, Jiang, CHENFANFU, and Sifakis, EFTYCHIOS. "An adaptive generalized interpolation material point method for simulating elastoplastic materials". *ACM Trans. Graph.* 36.6 (Nov. 2017), 1–12 7.

[HN81] Hirt, C W and Nichols, B D. "Volume of fluid (VOF) method for the dynamics of free boundaries". *J. Comput. Phys.* 39.1 (Jan. 1981), 201–225 3.

[Hou] Houdini, SIDEFX. *SideFX Houdini*. https://www.sidefx.com/products/houdini/. Accessed: 2024-9-27 3, 13.

[HSKF07] Hong, JEONG-MO, Shinar, TAMAR, Kang, MYUNGJOO, and Fedkiw, RONALD. "On boundary condition capturing for multiphase interfaces". en. *J. Sci. Comput.* 31.1-2 (May 2007), 99–125 3, 13.

[HW65] Harlow, FRANCIS H and Welch, J EDDIE. "Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface". en. *Phys. Fluids* 8.12 (Dec. 1965), 2182–2189 12.

[IBAT11] Ihmsen, MARKUS, Bader, JULIAN, Akinci, GIZEM, and Teschner, M. "Animation of air bubbles with SPH". *GRAPP* 2 (2011), 225–234 3.

[JS17] Jones, RICHARD and Southern, RICHARD. "Physically-based droplet interaction". en. *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. New York, NY, USA: ACM, July 2017 16.

[JSS*15] Jiang, CHENFANFU, Schroeder, CRAIG, Selle, ANDREW, et al. "The affine particle-in-cell method". *ACM Trans. Graph.* 34.4 (July 2015), 1–10 14.

[JWL*23] Jeske, STEFAN RHYS, Westhofen, LUKAS, Löschner, FABIAN, et al. "Implicit surface tension for SPH fluid simulation". en. *ACM Trans. Graph.* 1.1 (Nov. 2023) 3.

[Kim05] Kim, JUNSEOK. "A continuous surface tension force formulation for diffuse-interface models". en. *J. Comput. Phys.* 204.2 (Apr. 2005), 784–804 3.

[Kim12] Kim, JUNSEOK. "Phase-field models for multi-component fluid flows". en. *Commun. Comput. Phys.* 12.3 (Sept. 2012), 613–661 3.

[KLK20] Karnakov, PETR, Litvinov, SERGEY, and Koumoutsakos, PETROS. "A hybrid particle volume-of-fluid method for curvature estimation in multiphase flows". *Int. J. Multiphase Flow* 125 (Apr. 2020), 103209 3, 13.

[KLK22] Karnakov, PETR, Litvinov, SERGEY, and Koumoutsakos, PETROS. "Computing foaming flows across scales. From breaking waves to microfluidics". en. *Sci Adv* 8.5 (Feb. 2022), eabm0590 2.

[KLL*07] Kim, BYUNGMOON, Liu, YINGJIE, Llamas, IGNACIO, et al. "Simulation of bubbles in foam with the volume control method". *ACM Trans. Graph.* 26.3 (July 2007), 98–es 3.

[KSK10] Kim, DOYUB, Song, OH-YOUNG, and Ko, HYEONG-SEOK. "A practical simulation of dispersed bubble flow". *ACM SIGGRAPH 2010 papers*. SIGGRAPH '10 Article 70. New York, NY, USA: Association for Computing Machinery, July 2010, 1–5 2, 4.

[LLD*21] Li, WEI, Liu, DAOMING, Desbrun, MATHIEU, et al. "Kinetic-based multiphase flow simulation". en. *IEEE Trans. Vis. Comput. Graph.* 27.7 (July 2021), 3318–3334 3.

[LSD*22] Lesser, STEVE, Stomakhin, ALEXEY, Daviet, GILLES, et al. "Loki: a unified multiphysics simulation framework for production". *ACM Trans. Graph.* 41.4 (July 2022), 1–20 2–4, 7, 13, 16.

[LTKF08] Losasso, FRANK, Talton, JERRY, Kwatra, NIPUN, and Fedkiw, RONALD. "Two-way coupled SPH and particle level set fluid simulation". en. *IEEE Trans. Vis. Comput. Graph.* 14.4 (2008), 797–804 2, 4, 16.

[LWD24] Li, WEI, Wu, KUI, and Desbrun, MATHIEU. "Kinetic simulation of turbulent multifluid flows". en. *ACM Trans. Graph.* 43.4 (July 2024), 1–17 3.

[Man] Mantaflow. *Mantaflow*. http://mantaflow.com. Accessed: 2024-9-27 3.

[MBE*10] Misztal, MAREK KRZYSZTOF, Bridson, ROBERT, Erleben, KENNY, et al. *Optimization-based fluid simulation on unstructured meshes*. en. 2010 3.

[MCG03] Müller, MATTHIAS, Charypar, DAVID, and Gross, M. "Particle-based fluid simulation for interactive applications". *Int Conf Smart City Appl* (July 2003), 154–159 3, 5, 10.

[MEB*14] Misztal, MAREK KRZYSZTOF, Erleben, KENNY, Bargteil, ADAM, et al. "Multiphase flow of immiscible fluids on unstructured moving meshes". en. *IEEE Trans. Vis. Comput. Graph.* 20.1 (Jan. 2014), 4–16 3.

[OF02] Osher, STANLEY and Fedkiw, RONALD, eds. *Level set methods and dynamic implicit surfaces*. Springer My Copy UK, Oct. 2002 2.

[PAKF13] Patkar, SAKET, Aanjaneya, MRIDUL, Karpman, DMITRIY, and Fedkiw, RONALD. "A hybrid Lagrangian-Eulerian formulation for bubble generation and dynamics". *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '13. New York, NY, USA: Association for Computing Machinery, July 2013, 105–114 2, 4, 7.

[Pop09] Popinet, STÉPHANE. "An accurate adaptive solver for surface-tension-driven interfacial flows". *J. Comput. Phys.* 228.16 (Sept. 2009), 5838–5866 3, 13, 14.

[QLDJ22] Qu, ZIYIN, Li, MINCHEN, De Goes, FERNANDO, and Jiang, CHENFANFU. "The power particle-in-cell method". *ACM Trans. Graph.* 41.4 (July 2022), 1–13 8, 9.

[RJLL15] Ren, BO, Jiang, YUNTAO, Li, CHENFENG, and Lin, MING C. "A simple approach for bubble modelling from multiphase fluid simulation". en. *Comput. Vis. Media (Beijing)* 1.2 (June 2015), 171–181 4.

[RLY*14] Ren, BO, Li, CHENFENG, Yan, XIAO, et al. "Multiple-fluid SPH simulation using a mixture model". en. *ACM Trans. Graph.* 33.5 (Sept. 2014), 1–11 3.

[RLZ*21] Ruan, LIANGWANG, Liu, JINYUAN, Zhu, BO, et al. "Solid-fluid interaction with surface-tension-dominant contact". *ACM Trans. Graph.* 40.4 (July 2021), 1–12 3.

[SFK*08] Selle, ANDREW, Fedkiw, RONALD, Kim, BYUNGMOON, et al. "An Unconditionally Stable MacCormack Method". *J. Sci. Comput.* 35.2 (June 2008), 350–371 13.

[SLW*23] Stomakhin, ALEXEY, Lesser, STEVE, Wretborn, JOEL, et al. "Pahi: A Unified Water Pipeline and Toolset". *Proceedings of the Digital Production Symposium*. DigiPro '23 Article 11. New York, NY, USA: Association for Computing Machinery, Aug. 2023, 1–13 2, 3, 13, 16.

[SP08] Solenthaler, B and Pajarola, R. "Density contrast SPH interfaces". en. *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '08. Goslar, DEU: Eurographics Association, July 2008, 211–218 3.

[SWBD20] Stomakhin, ALEXEY, Wretborn, JOEL, Blom, KEVIN, and Daviet, GILLES. "Underwater bubbles and coupling". *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks*. New York, NY, USA: ACM, Aug. 2020 2–4, 7, 12, 13, 15.

[SZF12] Schroeder, CRAIG, Zheng, WEN, and Fedkiw, RONALD. "Semi-implicit surface tension formulation with a Lagrangian surface mesh on an Eulerian simulation grid". *J. Comput. Phys.* 231.4 (Feb. 2012), 2092–2115 3.

[TGK*17] Tampubolon, ANDRE PRADHANA, Gast, THEODORE, Klár, GERGELY, et al. "Multi-species simulation of porous sand and water mixtures". *ACM Trans. Graph.* 36.4 (July 2017), 1–11 2, 4.

[TSG15] Tripathi, MANOJ KUMAR, Sahu, KIRTI CHANDRA, and Govindarajan, RAMA. "Dynamics of an initially spherical bubble rising in quiescent liquid". en. *Nat. Commun.* 6 (Feb. 2015), 6268 2, 13.

[TWGT10] Thürey, NILS, Wojtan, CHRIS, Gross, MARKUS, and Turk, GREG. "A multiscale approach to mesh-based surface tension flows". *ACM Trans. Graph.* 29.4 (July 2010), 1–10 3.

[WFS22] Wretborn, JOEL, Flynn, SEAN, and Stomakhin, ALEXEY. "Guided bubbles and wet foam for realistic whitewater simulation". en. *ACM Trans. Graph.* 41.4 (July 2022), 1–16 2–4, 6–9, 14.

[WTGT09] Wojtan, CHRIS, Thürey, NILS, Gross, MARKUS, and Turk, GREG. "Deforming meshes that split and merge". *ACM SIGGRAPH 2009 papers*. SIGGRAPH '09 Article 76. New York, NY, USA: Association for Computing Machinery, July 2009, 1–10 3.

[YT13] Yu, JIHUN and Turk, GREG. "Reconstructing surfaces of particle-based fluids using anisotropic kernels". en. *ACM Trans. Graph.* 32.1 (Jan. 2013), 1–12 13.

[ZB05] Zhu, YONGNING and Bridson, ROBERT. "Animating sand as a fluid". *ACM Trans. Graph.* 24.3 (July 2005), 965–972 3, 14.

[ZSI*23] Zhai, XIAO, Schweickart, ESTON, Ilinov, NIKOLAY, et al. "Avatar: The Way of Hair, Cloth, and Coupled Simulation". *ACM SIGGRAPH 2023 Talks*. SIGGRAPH '23 Article 61. New York, NY, USA: Association for Computing Machinery, Aug. 2023, 1–2 7.

Appendix A: Equations of motion for an incompressible two-phase mixture

Here we show that the equations of motion Eqs. 1-3, together with conservation laws for density and fractional amounts, describe an incompressible two-phase mixture, and that the full set of equations describe a closed system.

We start with the bubble phase, which we for now assume is purely ballistic

$$\frac{D^b \mathbf{u}^b}{Dt} = \mathbf{g} \tag{52}$$

This is a closed system of 3 scalar equations (or 1 vector equation) with 3 components of the velocity field being the unknowns. We can further add the bubble mass conservation equation

$$\frac{\partial \phi^b \rho^b}{\partial t} + \nabla \cdot (\phi^b \rho^b \mathbf{u}^b) = 0, \tag{53}$$

and also the bubble fraction evolution equation

$$\frac{\partial \phi^b}{\partial t} + \nabla \cdot (\phi^b \mathbf{u}^b) = 0, \tag{54}$$

which implies conservation of bubble volume, or incompressibility of the bubble phase. The system 52, 53, 54 is closed as it contains 5 equations and 5 unknowns. We can also build the same exact system for water, and get a system of 10 equations with 10 unknowns

$$\frac{D^\star \mathbf{u}^\star}{Dt} = \mathbf{g}, \tag{55}$$

$$\frac{\partial \phi^\star \rho^\star}{\partial t} + \nabla \cdot (\phi^\star \rho^\star \mathbf{u}^\star) = 0, \tag{56}$$

$$\frac{\partial \phi^\star}{\partial t} + \nabla \cdot (\phi^\star \mathbf{u}^\star) = 0, \tag{57}$$

where $\star \in \{b, w\}$. Requiring fractions for bubbles and water to add up to 1, and adding a Lagrange multiplier in the form of the pore pressure, yields a system of 11 equations with 11 unknowns:

$$\phi^\star \rho^\star \frac{D^\star \mathbf{u}^\star}{Dt} = \phi^\star \rho^\star \mathbf{g} - \phi^\star \nabla p, \tag{58}$$

$$\frac{\partial \phi^\star \rho^\star}{\partial t} + \nabla \cdot (\phi^\star \rho^\star \mathbf{u}^\star) = 0, \tag{59}$$

$$\frac{\partial \phi^\star}{\partial t} + \nabla \cdot (\phi^\star \mathbf{u}^\star) = 0, \tag{60}$$

$$\phi^b + \phi^w = 1. \tag{61}$$

One could also add drag and surface tension force density contributions to the right-hand side of Eq. 58. Note that, Eq. 61 is equivalent to

$$\frac{\partial \phi^b}{\partial t} + \frac{\partial \phi^w}{\partial t} = 0, \quad (\phi^b + \phi^w)\Big|_{t=0} = 1, \tag{62}$$

which with the use of Eq. 60 becomes

$$\nabla \cdot (\phi^b \mathbf{u}^b + \phi^w \mathbf{u}^w) = 0, \quad (\phi^b + \phi^w)\Big|_{t=0} = 1,$$

matching the mixture incompressibility condition in Eq. 3.

In our case, since we use particles with assigned and preserved volumes to compute bubble fraction values on the grid (Eq. 30), it eliminates the need for Eq. 60. Likewise, using densities stored and preserved on particles to compute grid-based density values (Eq. 32), eliminates the need for Eq. 59. Ultimately, this leads to the system of Eqs. 1-3.

We make a final point: in mixture setups the material density of a phase, water or air, is the mass of the phase per unit volume *of that phase*. It is generally different from the mass of a phase per unit volume *of the world space* (since the phase may only partially occupy that volume), which can be obtained from the phase density via multiplication by its fraction. Unlike material densities force densities are defined per unit volume of the world space. In order to compute the force exerted per unit volume of a phase, the force density needs to be divided by its fraction.

Appendix B: Momentum conservation of drag force rasterization

Consider the drag force contribution from a single particle

$$\begin{aligned}
\sum_{\tau \in \mathcal{T}} \phi^w_\tau D_\tau V_\tau \mathbf{e}_\tau &\overset{\text{Eq. 39}}{=} \sum_{\tau \in \mathcal{T}} \phi^w_\tau \mathbf{e}_\tau (\mathbf{D}_q \cdot \mathbf{e}_\tau) V_q \omega_{q\tau} \\
&= V_q \sum_{\xi \in \{x,y,z\}} \left(\mathbf{D}_q \cdot \mathbf{e}_\xi\right) \sum_{\tau \in \mathcal{T}_\xi} \phi^w_\tau \mathbf{e}_\tau \omega_{q\tau} &(63)\\
&\overset{\text{Eq. 34}}{=} V_q \phi^w(\mathbf{x}_q) \mathbf{D}_q &(64)\\
&\overset{\text{Eq. 37}}{=} V_q \mathbf{d}^b_q.
\end{aligned}$$

On line 63 we can identify $\sum_{\tau \in \mathcal{T}_\xi} \phi^w_\tau \mathbf{e}_\xi \omega_{q\tau}$ as one component of the interpolation in Eq. 34. Combined with the component-wise multiplication on line 64 (same as in Eq. 37) we see that the proposed drag force rasterization scheme conserves the total drag force.

Appendix C: Momentum conservation of drift compensation

We consider the shifted surface tension force defined per face and pocket $\mathbf{f}^\sigma_\tau - \chi_\nu, \tau \in \mathcal{O}_\nu$ which by definition in Eqs. 43-44 integrates to zero over the pocket (Eq. 45).

We can also perform this integral on the particles. Assume that the all faces a particle q rasterize to belong to single pocket, that is $\forall \tau \in \mathcal{T} : \omega_{q,\tau} > 0 \implies \mathcal{O}_\nu$ for any pocket ν. This assumption is true for our choices of Λ_q. It makes sense to consider q as part of the pocket ν, which we will denote as $q \in \mathcal{P}_\nu$. Then the surface tension force integral on the particles is

$$\sum_{q \in \mathcal{P}_\nu} V_q f^\sigma(\mathbf{x}_q) \overset{\text{Eq. 46}}{=} \sum_{q \in \mathcal{P}_\nu} V_q \left(\sum_{\tau \in \mathcal{O}_\nu} \phi^c_\tau (f^\sigma_\tau - \chi_\nu) \omega_{q\tau} \right).$$

We can reorder the two sums and identify the bubble volume $\phi^b_\tau V_\tau$

$$\sum_{\tau \in \mathcal{O}_\nu} \left(\phi^c_\tau (f^\sigma_\tau - \chi_\nu) \sum_{q \in \mathcal{P}} V_q \omega_{q\tau} \right) \overset{\text{Eq. 31}}{=} \sum_{\tau \in \mathcal{O}_\nu} \phi^b_\tau V_\tau (f^\sigma_\tau - \chi_\nu).$$

Then,

$$\begin{aligned}
\mathbf{0} &\overset{\text{Eq. 45}}{=} \sum_{\tau \in \mathcal{O}_\nu} V_\tau \left(f^\sigma_\tau - \chi_\nu\right) \\
&= \sum_{\tau \in \mathcal{O}_\nu} \phi^b_\tau V_\tau \left(f^\sigma_\tau - \chi_\nu\right) + \sum_{\tau \in \mathcal{O}_\nu} \phi^w_\tau V_\tau \left(f^\sigma_\tau - \chi_\nu\right) &(65)\\
&= \sum_{q \in \mathcal{P}_\nu} V_q f^\sigma(\mathbf{x}_q) + \sum_{\tau \in \mathcal{O}_\nu} \phi^w_\tau V_\tau \left(f^\sigma_\tau - \chi_\nu\right). &(66)
\end{aligned}$$

That is, whether you integrate the shifted surface tension force on the grid (Eq. 65) or particles (Eq. 66) the force is momentum conserving when considering any pocket.