Guided Bubbles and Wet Foam for Realistic Whitewater Simulation

JOEL WRETBORN, Weta Digital, New Zealand
SEAN FLYNN, Weta Digital, New Zealand
ALEXEY STOMAKHIN, Weta Digital, New Zealand

We present a method for enhancing fluid simulations with realistic bubble and foam detail. We treat bubbles as discrete air particles, two-way coupled with a sparse volumetric Euler flow, as first suggested in [Stomakhin et al. 2020]. We elaborate further on their scheme and introduce a bubble inertia correction term for improved convergence. We also show how one can add bubbles to an already existing fluid simulation using our novel guiding technique, which performs local re-simulation of fluid to achieve more interesting bubble dynamics through coupling. As bubbles reach the surface, they are converted into foam and simulated separately. Our foam is discretized with smoothed particle hydrodynamics (SPH), and we replace forces normal to the fluid surface with a fluid surface manifold advection constraint to achieve more robust and stable results. The SPH forces are derived through proper constitutive modeling of an incompressible viscous liquid, and we explain why this choice is appropriate for "wet" types of foam. This allows us to produce believable dynamics from close-up scenarios to large oceans, with just a few parameters that work intuitively across a variety of scales. Additionally, we present relevant research on air entrainment metrics and bubble distributions that have been used in this work.

CCS Concepts: • **Computing methodologies** → **Physical simulation**; *Simulation types and techniques.*

Additional Key Words and Phrases: water, bubbles, foam, coupling

ACM Reference Format:
Joel Wretborn, Sean Flynn, and Alexey Stomakhin. 2022. Guided Bubbles and Wet Foam for Realistic Whitewater Simulation. *ACM Trans. Graph.* 41, 4, Article 117 (July 2022), 16 pages. https://doi.org/10.1145/3528223.3530059

1 INTRODUCTION

Grid-based Navier-Stokes simulators have a long history of producing impressive examples of realistic water behaviors in visual effects and animation. Unfortunately, they are inherently limited by the grid resolution which makes them impractical for capturing small-scale phenomena related to water aeration, such as tiny spray and mist droplets off of breaking waves, swarms of fine underwater bubbles, and intricate foam patterns on the surface. These simulators are typically used to capture mid- to large-scale motion, which is commonly referred to as *bulk* fluid.

Using adaptive grids may help capture extra surface detail, but it is still prohibitively expensive when it comes to Eulerian resolution for water-air interactions. Consequently, spray, mist, bubbles, and foam—or collectively *whitewater* effects—are usually simulated with Lagrangian particles. In a production environment it is often desirable to have independent control over the looks of bulk fluid and whitewater, so proper two-way coupling between them is almost never justified. Instead, whitewater particles are simulated in a separate, or *secondary*, pass on pre-cached bulk fluid data which remains unaffected.

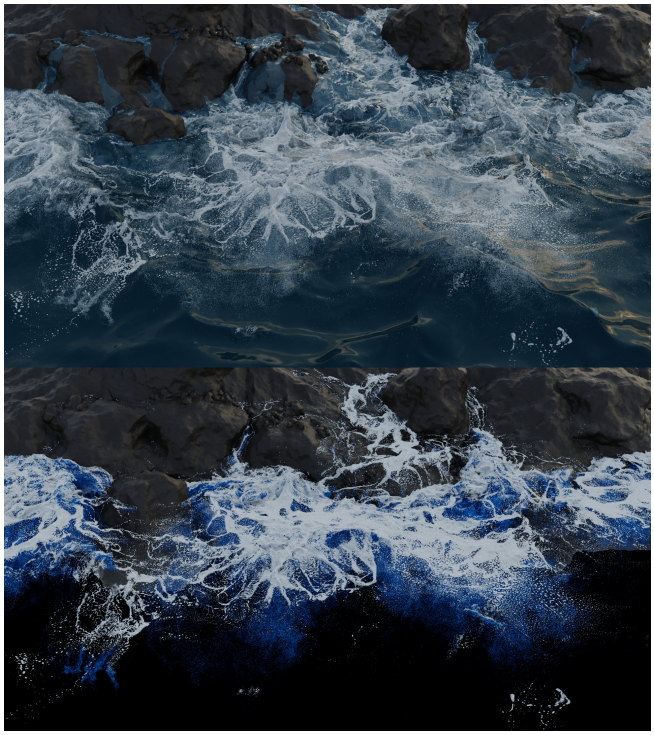

Fig. 1. **Rocky beach.** Ocean waves splash against a rocky beach creating fascinating bubble and foam patterns simulated using our method. The bottom view displays the bubbles (blue) and foam (white) more clearly without the water surface. ©Wētā FX.

Existing secondary simulation techniques are mostly phenomenological: a collection of simple models is embedded into a large and often overfitted parameter space, requiring fine tuning of emission and dynamics for each specific scenario. This paradigm puts a large burden and reliance on highly trained technical artists. Using these techniques, they do achieve remarkable results; but because of the vast art-directable space, it is easy to exit the realm of believability without a clear direction of how to get back. These issues manifest in VFX production with comments from artists and directors such as "the whitewater does not seem connected to the fluid" or "I don't believe <insert secondary effect> really behaves that way".

We address these issues by presenting new methods for simulating whitewater effects in a more principled manner. We focus

All images ©Wētā FX.

Authors' addresses: Joel Wretborn, Weta Digital, New Zealand, joel@wbn.se; Sean Flynn, Weta Digital, New Zealand, sflynn@wetafx.co.nz; Alexey Stomakhin, Weta Digital, New Zealand, st.alexey@gmail.com.

0730-0301/2022/7-ART117 $15.00
https://doi.org/10.1145/3528223.3530059

on the whitewater components consisting of entrapped air, namely underwater bubbles and surface foam. Our technique can produce impressive whitewater effects that appear naturally connected to the fluid in complex scenarios such as waves breaking against a rocky beach (Figure 1) or a submarine breaching the ocean surface (Figure 2). While our approach is still primarily intended as a secondary technique, our methods are versatile; and running them simultaneously with the bulk water simulation in certain scenarios can yield equally impressive results.

Previous bubble simulation techniques such as that of [Bender et al. 2019; Ihmsen et al. 2012] simply advect bubble particles through the containing fluid velocity field. One-way drag and buoyancy forces may also be applied. While this does give the bubbles some interesting motion, it lacks the two-way interaction that is essential for the complex dynamics observed in the real world. In particular, having bubbles affect the dynamics of water is essential to capture *collective* effects, such as a collection of bubbles rising faster than a single bubble, or dense groups of bubbles rising faster than sparse ones. We build upon the idea from [Stomakhin et al. 2020] to achieve this interaction, which provides a much more realistic, connected look, and propose a bubble *inertia correction* to improve convergence of the coupling scheme. Since the method is based on discretizing the equations of continuum mechanics, bubble dynamics look plausible out of the box, requiring little to no parameter tuning.

We then introduce our *guided* simulation approach, which couples bubbles with a locally re-simulated fluid, driven by a pre-cached bulk motion. This way we maintain the interesting dynamics of a two-way coupled scheme, while effectively running it as a secondary simulation, thus preserving the look of the original bulk fluid.

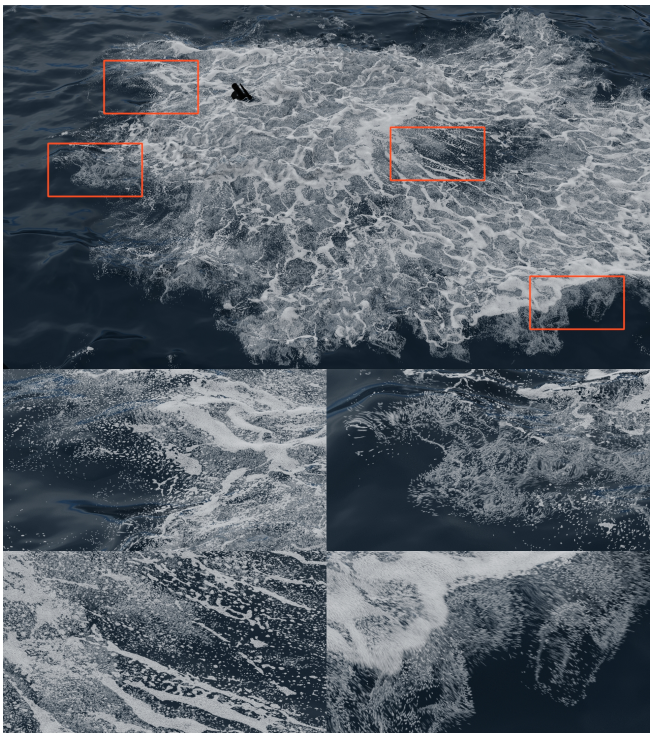

Fig. 2. **Submarine.** A massive wake from a submarine creates swirling bubbles and intricate foam patterns. ©Wētā FX.

Previous foam simulation techniques have either been overly simplistic with particles merely being projected to the fluid surface as they are advected [Ihmsen et al. 2012], or too complex with expensive solves that are impractical for large ocean scenarios [Busaryev et al. 2012; Yue et al. 2015]. We take a principled route and use a constitutive model for "wet" foam which we discretize using smoothed particle hydrodynamics (SPH). The model has an intuitive parameter space and produces believable results across a range of scales. We also provide a *manifold advection* technique that robustly connects our foam with the liquid surface, based on [Morgenroth et al. 2020]. Our foam simulation approach has the fidelity for close-ups while it also scales well and is able to cover large areas.

To determine aerated regions of water for bubble emission we employ a simple *aeration* heuristic based on [Gualtieri et al. 2008], Chapter 7. The bubbles are then emitted using an inverse-cubic probability density function found in physics literature [Deike et al. 2016]. The overall procedure is controlled by a small and intuitive parameter set and tends to consistently produce believable bubble distributions. We also discuss our strategy for transitioning bubbles into foam as they reach the water surface.

To summarize, our contributions are:

- Stable coupling scheme for bubbles and water.
- Guided bubbles simulation driven by a pre-cached bulk fluid.
- SPH discretization of a "wet" foam constitutive model.
- Manifold advection for precise surface tracking of foam.
- Emission and transition heuristics for bubbles and foam.

The combined result from these contributions creates a state-of-the-art bubble and foam solver that we believe improves on the currently most widely used systems.

The paper continues with a discussion of previous work in Section 2. We introduce the coupling scheme between bubbles and water in Section 3, followed by our novel guiding technique in Section 4 and emission heuristics in Section 5. In Section 6 we present our foam model, followed by Section 7 where we describe our state transition strategies. Finally, we discuss results and conclusions in Sections 8 and 9.

2 RELATED WORK

Our guided bubbles and foam methods augment fluid simulations generated using modern computer graphics techniques that began with the work of [Stam 1999]. It was extended with a particle level set method from [Enright et al. 2002] to support accurate fluid-air interface tracking and enable convincing water simulations. We specifically focus on applying our method to Fluid Implicit Particle (FLIP) [Zhu and Bridson 2005] simulations, as it is the state-of-the-art bulk water simulation technique in VFX production implemented in such software packages as SideFX Houdini and Autodesk Bifröst. These and other fluid simulation techniques are covered in detail in [Bridson 2015]. Other notable improvements in water simulation include reduced dissipation through the use of higher order particle-grid transfer methods [Fu et al. 2017; Jiang et al. 2015] and adaptive techniques [Ando and Batty 2020; Losasso et al. 2004].

Table 1. Summary of notations used to describe our coupling scheme between water and bubbles. Locations are p (particle), f (MAC face center), and c (MAC cell center).

Symbol	Units	Location	Meaning
ϕ_b	1	f	volume fraction of bubbles
ϕ_w	1	f	volume fraction of water
$\boldsymbol{b}$	N/m^3	f	buoyancy force density
$\boldsymbol{f}$	N/m^3	f	drag force density
$\boldsymbol{v}$	m/s	f	bubble velocity field
$\boldsymbol{u}$	m/s	f	water velocity field
P	Pa	c	(pore) pressure
∇P	Pa/m	f	pressure gradient
$\boldsymbol{x}_p$	m	p	position of particle p
$\boldsymbol{v}_p$	m/s	p	velocity of particle p
m_p	kg	p	mass of particle p
r_p	m	p	radius of particle p
V_p	m^3	p	volume of particle p
$\boldsymbol{f}_p$	N	p	drag force on particle p
$(\nabla P)_p$	Pa/m	p	pressure gradient at $\boldsymbol{x}_p$

A number of research papers focus specifically on modeling underwater bubbles. Kim et al. [2010] passively advect bubble particles with the bulk fluid and use them to adjust effective density of water, leading to naturalistic collective buoyancy effects. Their method relies on a so-called Boussinesq approximation, which locks bubble motion to that of water. In order to break up the bubble dynamics they employ an ad hoc stochastic solver. Patkar et al. [2013] use an Eulerian two-phase approach for simulating bubbles larger than the fluid grid voxel size and passively advected particles for tracking bubbles smaller than the fluid grid voxel size. They combine the two in a single linear solve which also handles compressibility. The technique works great for fully coupled close-up scenarios where dynamics of the Eulerian counterpart motion can be fully appreciated, but it would not scale to large ocean setups. Their Lagrangian bubble counterpart on the other hand is simply driven by the bulk water unidirectionally, similar to [Ihmsen et al. 2012]. Hong et al. [2008] consider two-way coupling of Lagrangian bubbles with the surrounding Eulerian fluid. The coupling scheme however is based on an explicit exchange of lift, drag, and buoyancy forces, leading to restrictive timestep requirements. The forces are also merely defined heuristically, potentially making it difficult to consistently achieve a desired look in a production environment. Losasso et al. [2008] also recognize the importance of two-way coupling simulating Eulerian bulk water together with an aerated SPH component. We were primarily inspired by the work of [Stomakhin et al. 2020], which introduces a robust, semi-implicit method for two-way coupling a grid-based fluid (Eulerian or FLIP) with Lagrangian bubble particles through physically based drag and pore pressure buoyancy forces, and expand on their work by introducing a more stable integration scheme and a novel guiding approach. Similar coupling schemes have also been explored in other contexts such as modeling grains, hair, and cloth interacting with a fluid [Daviet and Bertails-Descoubes 2017; Fei et al. 2018; Gao et al. 2018; Tampubolon et al. 2017].

Foam is a hard phenomenon to simulate accurately, and most of the existing physically based methods would have difficulty scaling to cover large areas with foam. Busaryev et al. [2012] produce high-fidelity results by simulating each bubble-bubble interaction using Voronoi cells. Yue et al. [2015] consider foam consisting only of tiny bubbles and employ an elasto-plastic continuous model which they discretize using a material point method (MPM). Cleary et al. [2007] use SPH to simulate the formation of thick foam in carbonated liquids, and employ a soft particle discrete element style method to model collisions between individual bubbles. For our application we sought to prioritize scaling and interaction with the water surface over the accuracy of the foam model itself, and resorted to a simpler "wet" foam model which we present later. We use SPH [Becker and Teschner 2007; Monaghan 1992; Müller et al. 2003] to discretize our foam. Modeling the relevant surface tension and cohesion effects with SPH is covered in [Akinci et al. 2013; Clavet et al. 2005; He et al. 2014], and viscosity treatment is discussed in [Peer et al. 2015]. Dehnen and Aly [2012] give an overview of different SPH kernels.

Table 2. Summary of the used physical constants.

Symbol	Units	Value	Meaning
$\boldsymbol{g}$	m/s^2	$(0, -9.81, 0)$	gravity vector
ρ_w	kg/m^3	1000	density of water
ρ_b	kg/m^3	1	density of air
μ_w	Pa·s	0.001	dynamic viscosity of water
γ	N/m	0.072	surface tension of water

The literature on complete whitewater systems is rather sparse. The first detailed overview has been presented in [Ihmsen et al. 2012], with minor modifications later suggested in [Bender et al. 2019]. The SideFX Houdini whitewater solver seems to mostly follow [Ihmsen et al. 2012], but also employs a PBD solver for the foam to maintain distances between its particles. There also appears to be little to no consensus on the optimal bubble emission metric. Artists typically tune some combination of bulk fluid vorticity, velocity magnitude and surface mean curvature, see [Bender et al. 2019; Ihmsen et al. 2012; Patkar et al. 2013]. Due to the manual and phenomenological nature of the process, it tends to require a lot of artistic time and does not translate easily from one setup to another. We thus turn to more physically based emission metrics such as the ones outlined in [Deike et al. 2016; Gualtieri et al. 2008].

3 COUPLING BUBBLES AND WATER

We begin by describing our approach for simulating air bubbles entrapped in water. Our method is two-way coupled to capture the essential interaction between bubbles and water that is absent in previous methods. Using this method artists can easily achieve a natural, connected look with less time-consuming parameter tuning. Table 1 lists the notations used in this section, and Table 2 provides a summary of physical constants used throughout this work.

3.1 Governing equations

To model interactions between bubbles and the surrounding fluid we adopt a continuum approach. Let ϕ_b and $\phi_w = 1 - \phi_b$ denote the volume fractions of bubbles and water respectively. Following [Anderson and Jackson 1967; Daviet and Bertails-Descoubes 2017], we write the internal pressure of the water and bubbles as fractions of the pore pressure P, that is $\phi_b P$ and $\phi_w P$, respectively. The force per unit volume exerted by the water onto the bubbles is the sum

of drag $\boldsymbol{f}$, discussed later, and generalized buoyancy

$$\boldsymbol{b} = \phi_w \nabla (\phi_b P) - \phi_b \nabla (\phi_w P).$$

The conservation of momentum equations for bubble and water phases thus read

$$\phi_b \rho_b \frac{D\boldsymbol{v}}{Dt} = \phi_b \rho_b \boldsymbol{g} + \boldsymbol{f} + \boldsymbol{b} - \nabla (\phi_b P),$$
$$\phi_w \rho_w \frac{D\boldsymbol{u}}{Dt} = \phi_w \rho_w \boldsymbol{g} - \boldsymbol{f} - \boldsymbol{b} - \nabla (\phi_w P),$$

respectively, where $\frac{D}{Dt}$ is the material derivative, $\boldsymbol{v}$ is the velocity of bubbles, $\boldsymbol{u}$ is the velocity of water, ρ_b is the density of air, ρ_w is the density of water, and $\boldsymbol{g}$ is gravity. Noting that $\boldsymbol{b} = P\nabla\phi_b = -P\nabla\phi_w$, the momentum equations can be rewritten as

$$\phi_b \rho_b \frac{D\boldsymbol{v}}{Dt} = \phi_b \rho_b \boldsymbol{g} + \boldsymbol{f} - \phi_b \nabla P \tag{1}$$

$$\phi_w \rho_w \frac{D\boldsymbol{u}}{Dt} = \phi_w \rho_w \boldsymbol{g} - \boldsymbol{f} - \phi_w \nabla P. \tag{2}$$

The system is closed by enforcing incompressibility of the mixture

$$\nabla \cdot (\phi_w \boldsymbol{u} + \phi_b \boldsymbol{v}) = 0. \tag{3}$$

3.2 Spatial discretization

Bubbles. We use Lagrangian particles to represent the bubbles. Assuming a non-zero bubble fraction ϕ_b and integrating the bubbles conservation equation (1) over the volume V_p of a particle yields

$$m_p \frac{\mathrm{d}\boldsymbol{v}_p}{\mathrm{d}t} = m_p \boldsymbol{g} + \frac{V_p}{\phi_b} \boldsymbol{f}(\boldsymbol{x}_p) - V_p \nabla P(\boldsymbol{x}_p), \quad \frac{\mathrm{d}\boldsymbol{x}_p}{\mathrm{d}t} = \boldsymbol{v}_p, \tag{4}$$

where $m_p = \rho_b V_p$, $\boldsymbol{v}_p$ and $\boldsymbol{x}_p$ are mass, velocity, and position of the bubble particle, respectively. Instead of starting with a drag force density field $\boldsymbol{f}$, we define the drag force per particle $\boldsymbol{f}_p = \frac{V_p}{\phi_b} \boldsymbol{f}(\boldsymbol{x}_p)$ in equation (4) directly as

$$\boldsymbol{f}_p = \chi_b \mu_w r_p \left(6\pi + \frac{\pi}{2} \frac{\rho_w r_p \|\Delta \boldsymbol{v}_p\|}{\mu_w}\right) \Delta \boldsymbol{v}_p. \tag{5}$$

We can then retrieve force density $\boldsymbol{f}$ by splatting $\boldsymbol{f}_p$ onto the Eulerian grid to be used in the water equation (2). Here $\Delta \boldsymbol{v}_p = \boldsymbol{v}_p - \boldsymbol{u}(\boldsymbol{x}_p)$ is the difference in velocity between the bubble and the surrounding fluid, μ_w is the dynamic viscosity of water, r_p is the radius of the bubble, and χ_b is a dimensionless bubble drag coefficient.

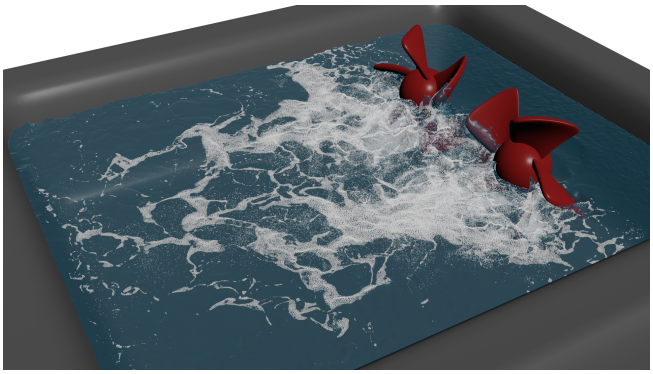

Fig. 3. **Propellers.** Two rotating propellers induce fluid circulation inside of a pool making naturalistic bubble and foam patterns. ©Wētā FX.

Equation (5) was borrowed from [Daviet and Bertails-Descoubes 2017]; it combines the linear Stokes drag force formula for laminar flows with a second-order term that captures turbulent bubble-water interactions. The bubble motion equations (4) then become

$$m_p \frac{\mathrm{d}\boldsymbol{v}_p}{\mathrm{d}t} = m_p \boldsymbol{g} + \boldsymbol{f}_p - V_p \nabla P(\boldsymbol{x}_p), \quad \frac{\mathrm{d}\boldsymbol{x}_p}{\mathrm{d}t} = \boldsymbol{v}_p. \tag{6}$$

Water. We use standard staggered marker-and-cell (MAC) grid discretization for the water [Harlow and Welch 1965]. Even though we mainly focus on fully Eulerian fluids in our derivations, it is worth noting that our coupling approach works exactly the same way for FLIP fluids, since they have a grid-based representation.

Fields $\boldsymbol{u}$, ρ_w, ϕ_w, and ∇P are defined on grid faces, while P is defined at cell centers. The drag force $\boldsymbol{f}_p$ and buoyancy are calculated on the bubbles by interpolating surrounding water velocity $\boldsymbol{u}$ and pressure gradient ∇P from the Eulerian grid. $\boldsymbol{f}_p$ is then rasterized back to the faces of the grid with a minus sign, and added as a forcing term $-\boldsymbol{f}$ to the bulk fluid; so equation (2) becomes

$$\rho_w \frac{D\boldsymbol{u}}{Dt} = \rho_w \boldsymbol{g} - \frac{\boldsymbol{f}}{\phi_w} - \nabla P. \tag{7}$$

Fractions ϕ_b are computed by rasterizing bubble volumes to grid faces. We then compute $\phi_w = 1 - \phi_b$. Bubble velocities $\boldsymbol{v}_p$ are also splatted to grid faces to obtain $\boldsymbol{v}$. Rasterization and interpolation are described in detail in Section 3.3.1. We next discuss numerical integration of the system of equations (6), (7), and (3).

3.3 Time integration

We do a standard operator splitting of the self-advection term in the fluid momentum equation (7), allowing us to advect both water and bubble particles separate from the rest of the solve at the beginning of each timestep. With advection separated, the water velocity update equation becomes

$$\rho_w \frac{\partial \boldsymbol{u}}{\partial t} = \rho_w \boldsymbol{g} - \frac{\boldsymbol{f}}{\phi_w} - \nabla P. \tag{8}$$

For the remainder of the timestep the positions of water and bubble particles are assumed to be fixed, and only velocities are updated. This way we ensure consistent treatment of advection and velocity updates for both phases.

Within each Newton step we solve for the velocities of bubbles using the first equation in (6), assuming the prescribed water velocity and pressure. We then solve (8) and (3) for the water by performing a pressure projection, assuming the prescribed velocities of the bubbles. As mentioned earlier, since the bubbles are modeled as Lagrangian particles interacting with an Eulerian bulk fluid through buoyancy and drag, additional splatting and interpolation is required to transfer particle data to and from the grid. An overview of our scheme is shown in Figure 4.

3.3.1 Rasterization and interpolation. We use $(\boldsymbol{i}, \alpha)$ to refer to a face of an Eulerian grid, indexed by voxel $\boldsymbol{i}$ and axis α. The value at face $(\boldsymbol{i}, \alpha)$ of the bubble fraction ϕ_b field is computed by accumulating weighted particle volumes and then dividing by the volume of a grid voxel (stage 3 in Figure 4)

$$(\phi_b)_{\boldsymbol{i},\alpha} = \frac{\sum_p V_p N_p^{\boldsymbol{i},\alpha}}{(\Delta x)^3}, \tag{9}$$

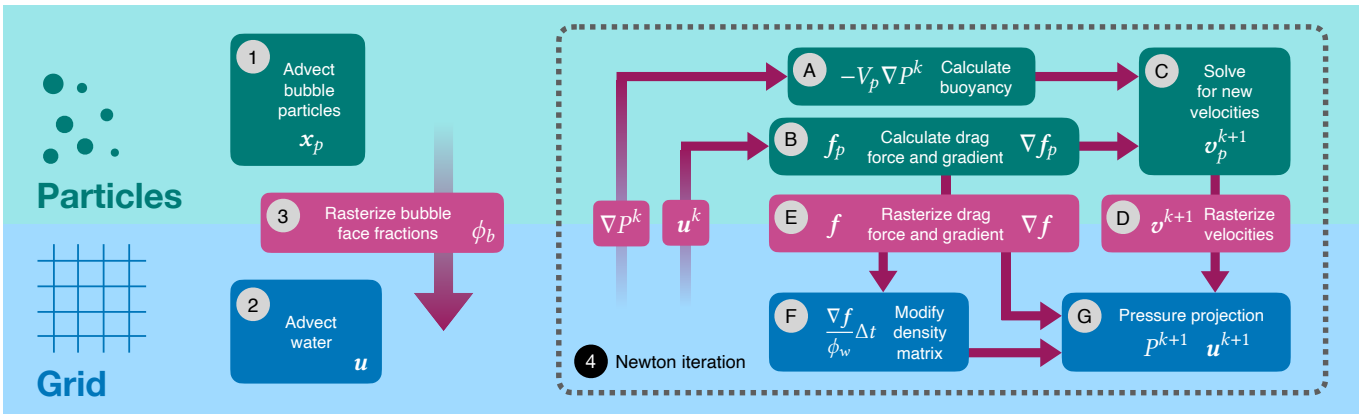

Fig. 4. Our scheme for coupling Lagrangian bubble particles (top) with an Eulerian fluid (bottom). For every timestep, stages 1-4 are executed in order. Stages A-G are executed in order for each Newton step to obtain the next set of approximations $\boldsymbol{v}_p^{k+1}$, P^{k+1}, and $\boldsymbol{u}^{k+1}$. ©Wētā FX.

where $N_p^{\boldsymbol{i},\alpha}$ is the tri-linear weight of a particle p with respect to the center of the face $(\boldsymbol{i},\alpha)$, and Δx is the voxel size of the Eulerian grid. Bubble velocities $\boldsymbol{v}_p$ are rasterized using a momentum-conserving MPM-style [Stomakhin et al. 2013] method, by accumulating volume-weighted velocities first and then dividing by the total volume

$$v_{\boldsymbol{i},\alpha} = \frac{\sum_p v_p^\alpha V_p N_p^{\boldsymbol{i},\alpha}}{\sum_p V_p N_p^{\boldsymbol{i},\alpha}}. \tag{10}$$

Here v_p^α denotes the component α of vector $\boldsymbol{v}_p$, that is $v_p^\alpha = \boldsymbol{v}_p \cdot \boldsymbol{e}_\alpha$, where $\boldsymbol{e}_\alpha$ is the unit vector along axis α. Note that face $(\boldsymbol{i},\alpha)$ velocity $v_{\boldsymbol{i},\alpha}$ is a scalar, as are all other quantities defined on MAC grid faces. Unlike fractions ϕ_b, velocities $\boldsymbol{v}$ need to be recomputed through rasterization for every Newton step to mirror the updates on bubble velocities $\boldsymbol{v}_p$ (stage D of Figure 4). Particle drag forces $\boldsymbol{f}_p$ are rasterized to grid faces (stage E of Figure 4) to obtain the grid force density $\boldsymbol{f}$ as

$$f_{\boldsymbol{i},\alpha} = \frac{\sum_p f_p^\alpha N_p^{\boldsymbol{i},\alpha}}{(\Delta x)^3}. \tag{11}$$

The pressure gradient ∇P and velocity $\boldsymbol{u}$ for stages A and B of Figure 4 are interpolated to particles, respectively, as

$$(\nabla P)_p = \sum_\alpha \sum_{\boldsymbol{i}} (\nabla P)_{\boldsymbol{i},\alpha} N_p^{\boldsymbol{i},\alpha} \boldsymbol{e}_\alpha, \tag{12}$$

$$\boldsymbol{u}_p = \sum_\alpha \sum_{\boldsymbol{i}} u_{\boldsymbol{i},\alpha} N_p^{\boldsymbol{i},\alpha} \boldsymbol{e}_\alpha. \tag{13}$$

3.3.2 Bubble velocity update. The bubble particle velocity update (stage C of Figure 4) happens in accordance with the first equation in (6). Performing a backward Euler Newton step on this ODE is straightforward, as the non-linearity only comes from the implicit dependence of $\boldsymbol{f}_p$ on $\boldsymbol{v}_p$

$$m_p \frac{\boldsymbol{v}_p^{k+1} - \boldsymbol{v}_p^0}{\Delta t} = m_p \boldsymbol{g} - V_p \nabla P^k(\boldsymbol{x}_p^0) + \boldsymbol{f}_p(\boldsymbol{v}_p^k) + \nabla_{\boldsymbol{v}} \boldsymbol{f}_p(\boldsymbol{v}_p^k)(\boldsymbol{v}_p^{k+1} - \boldsymbol{v}_p^k),$$

where superscript $k+1$ indicates the updated Newton estimate to be computed given the previous one with superscript k. $\boldsymbol{v}_p^0$ and $\boldsymbol{x}_p^0$ are the velocity and position of a bubble before the start of Newton iteration respectively, corresponding to $k = 0$, and Δt is the timestep size. $\boldsymbol{f}_p$ of course also depends on the surrounding fluid velocity, but we drop the $\boldsymbol{u}^k$ argument here for brevity as it is fixed during the bubble update. Rearranging the terms gives

$$\left(\frac{m_p}{\Delta t} - \nabla_{\boldsymbol{v}} \boldsymbol{f}_p(\boldsymbol{v}_p^k)\right)(\boldsymbol{v}_p^{k+1} - \boldsymbol{v}_p^k) = -\frac{m_p}{\Delta t}(\boldsymbol{v}_p^k - \boldsymbol{v}_p^0) + m_p \boldsymbol{g} + \boldsymbol{f}_p(\boldsymbol{v}_p^k) - V_p \nabla P^k(\boldsymbol{x}_p^0),$$

revealing that the velocity correction $\boldsymbol{v}_p^{k+1} - \boldsymbol{v}_p^k$ is computed by inverting a 3x3 matrix $\frac{m_p}{\Delta t} - \nabla_{\boldsymbol{v}} \boldsymbol{f}_p(\boldsymbol{v}_p^k)$.

Pressure gradient ∇P interpolated from the grid is not available at the start of Newton iteration. Though it can be obtained through advection from the previous timestep, we have found that using a zero pressure gradient for the first Newton step works as well.

3.3.3 Water velocity update. Computing pressure and updating fluid velocity on the Eulerian grid (stage G of Figure 4) happens in accordance with equation (8) coupled with the incompressibility condition (3). A backward Euler Newton step for (8) reads

$$\rho_w \frac{\boldsymbol{u}^{k+1} - \boldsymbol{u}^0}{\Delta t} = \rho_w \boldsymbol{g} - \frac{\boldsymbol{f}(\boldsymbol{u}^k)}{\phi_w} - \frac{\nabla_{\boldsymbol{u}} \boldsymbol{f}(\boldsymbol{u}^k)}{\phi_w}(\boldsymbol{u}^{k+1} - \boldsymbol{u}^k) - \nabla P^{k+1},$$

where $\boldsymbol{u}^0$ is the velocity of the fluid before the start of the Newton iteration. Here we have accounted for the fact that drag force $\boldsymbol{f}$ depends on fluid velocity $\boldsymbol{u}$, and hence the corresponding gradient is needed for the implicit update. In practice both $\boldsymbol{f}_p$ and $\nabla_{\boldsymbol{u}} \boldsymbol{f}_p$ are computed on the particles and then splatted with a negative sign (stage E of Figure 4) to the fluid grid faces, to get $-\boldsymbol{f}$ and $-\nabla_{\boldsymbol{u}} \boldsymbol{f}$ respectively. We adopt a diagonally lumped form of $\nabla_{\boldsymbol{u}} \boldsymbol{f}_p$ which makes splatting to a staggered MAC grid trivial: we simply splat the diagonal of $\nabla_{\boldsymbol{u}} \boldsymbol{f}_p$ to grid faces in the same manner that we splat $\boldsymbol{f}_p$ in (11). Rearranging the terms and combining with incompressibility condition (3) yields a system with respect to $\boldsymbol{u}^{k+1}$ and P^{k+1}

$$\left(\frac{\rho_w}{\Delta t} + \frac{\nabla_{\boldsymbol{u}} \boldsymbol{f}(\boldsymbol{u}^k)}{\phi_w}\right)(\boldsymbol{u}^{k+1} - \boldsymbol{u}^k) = -\frac{\rho_w}{\Delta t}(\boldsymbol{u}^k - \boldsymbol{u}^0) + \rho_w \boldsymbol{g} - \frac{\boldsymbol{f}(\boldsymbol{u}^k)}{\phi_w} - \nabla P^{k+1}, \tag{14}$$

$$\nabla \cdot \left(\phi_w \boldsymbol{u}^{k+1} + \phi_b \boldsymbol{v}^{k+1}\right) = 0. \tag{15}$$

The system of (14) and (15) is similar to a standard pressure projection for incompressible fluids, and we solve it using the variational framework of [Batty et al. 2007]. The difference is only in an extra density correction term $\frac{\nabla_{\boldsymbol{u}} \boldsymbol{f}(\boldsymbol{u}^k)}{\phi_w}\Delta t$, which is trivial to incorporate by adding it to ρ_w (stage D of Figure 4). This step effectively implies implicit integration of the drag force, which is important for faster convergence as we employ a non-linear drag formulation (5). Bubble fractions ϕ_b and velocities $\boldsymbol{v}^{k+1}$, obtained through splatting (stages 3 and D of Figure 4), simply act as kinematic colliders. We discuss the resulting discrete Poisson problem in detail in the next section, where we also introduce our bubble inertia correction method.

3.4 Pressure projection

Let us first derive the discrete Poisson problem corresponding to the system of equations (14) and (15). To simplify the notations, we define effective mass density and effective force density as

$$\rho_\star = \rho_w + \frac{\nabla_{\boldsymbol{u}} \boldsymbol{f}(\boldsymbol{u}^k)}{\phi_w}\Delta t, \quad \boldsymbol{f}_\star = -\frac{\rho_w}{\Delta t}\left(\boldsymbol{u}^k - \boldsymbol{u}^0\right) + \rho_w \boldsymbol{g} - \frac{\boldsymbol{f}(\boldsymbol{u}^k)}{\phi_w},$$

respectively, both of which would exist on the faces of the Eulerian grid. We also drop the $k+1$ index from the unknowns $\boldsymbol{u}$ and P we are solving for. With this the system becomes

$$\frac{\rho_\star}{\Delta t}\left(\boldsymbol{u} - \boldsymbol{u}^k\right) = \boldsymbol{f}_\star - \nabla P, \quad \nabla \cdot \left(\phi_w \boldsymbol{u} + \phi_b \boldsymbol{v}^{k+1}\right) = 0. \tag{16}$$

Let $\boldsymbol{G}$ be the discrete gradient operator which maps a field defined on grid cells to one defined on faces. Note that $-\boldsymbol{G}^T$ is then the corresponding discrete divergence operator. Also let $\boldsymbol{\Phi}_w$, $\boldsymbol{\Phi}_b$, $\boldsymbol{D}_\star$ be discrete diagonal operators defined on faces corresponding to ϕ_w, ϕ_b, and $\rho_\star$ respectively. The discrete representation of (16) is then

$$\frac{1}{\Delta t}\boldsymbol{D}_\star\left(\boldsymbol{u} - \boldsymbol{u}^k\right) = \boldsymbol{f}_\star - \boldsymbol{G}P, \quad \boldsymbol{G}^T\left(\boldsymbol{\Phi}_w \boldsymbol{u} + \boldsymbol{\Phi}_b \boldsymbol{v}^{k+1}\right) = \boldsymbol{0}, \tag{17}$$

where we slightly abuse the notations by reusing $\boldsymbol{u}$, $\boldsymbol{v}^{k+1}$, and $\boldsymbol{f}_\star$ as discrete vectors defined on grid faces, and P as a discrete vector defined on grid cells. Applying $\boldsymbol{G}^T\boldsymbol{\Phi}_w\boldsymbol{D}_\star^{-1}\Delta t$ to the first equation and substituting $\boldsymbol{G}^T\boldsymbol{\Phi}_w\boldsymbol{u}$ into the second one yields the Schur complement system for P

$$\Delta t \boldsymbol{G}^T\boldsymbol{\Phi}_w\boldsymbol{D}_\star^{-1}\boldsymbol{G}P = \boldsymbol{G}^T\left(\boldsymbol{\Phi}_w\left(\boldsymbol{u}^k + \Delta t \boldsymbol{D}_\star^{-1}\boldsymbol{f}_\star\right) + \boldsymbol{\Phi}_b\boldsymbol{v}^{k+1}\right), \tag{18}$$

which is a discrete Poisson problem, with a diagonally weighted discrete Laplacian for the matrix. The right hand side is the divergence of the mixture of "explicitly" updated fluid velocity $\boldsymbol{u}^k + \Delta t\boldsymbol{D}_\star^{-1}\boldsymbol{f}_\star$ and bubble velocity $\boldsymbol{v}^{k+1}$ in accordance with fractions $\boldsymbol{\Phi}_w$ and $\boldsymbol{\Phi}_b$. System (18) can be solved using a (preconditioned) conjugate gradient method. With the solution for P available, $\boldsymbol{u}$ is readily computed from the first equation in (17).

System of equations (16) has an obvious drawback: it treats bubbles as having a prescribed velocity, or being infinitely heavy. Given that the density of air is about 1000x smaller than that of water, this is a rather crude approximation. Even though our coupling scheme tends to produce good-looking results in practice, we did observe some cases where the Newton iteration scheme was having convergence issues; see Figure 6 (left). To fix the problem, we allow bubble velocity to participate in the solve by adding a reduced bubble momentum update equation to (16), to obtain

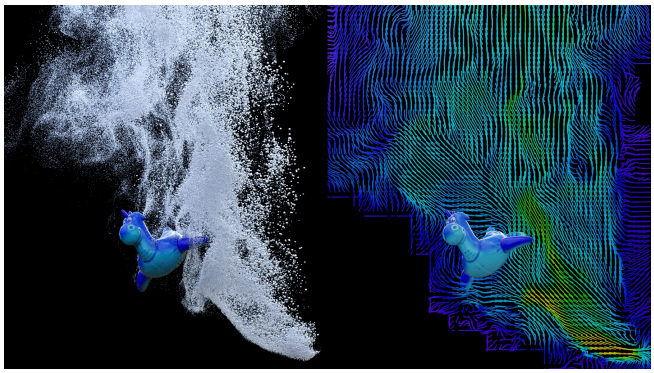

Fig. 5. **Coupling.** Lagrangian bubbles (left) two-way coupled with an Eulerian fluid (right) interacting with an animated toy. ©Wētā FX.

$$\frac{\rho_\star}{\Delta t}\left(\boldsymbol{u} - \boldsymbol{u}^k\right) = \boldsymbol{f}_\star - \nabla P, \tag{19}$$

$$\frac{\rho_b}{\Delta t}\left(\boldsymbol{v} - \tilde{\boldsymbol{v}}^{k+1}\right) = -\theta \nabla P, \tag{20}$$

$$\nabla \cdot \left(\phi_w \boldsymbol{u} + \phi_b \boldsymbol{v}\right) = 0. \tag{21}$$

We call the bubble momentum update (20) reduced because it only contains a buoyancy contribution, but not other forces such as drag. This is sufficient to account for the inertia of bubbles in the pressure projection. System of equations (19), (20), and (21) is to be solved for unknowns $\boldsymbol{u}$, $\boldsymbol{v}$, and P. Note how the incompressibility condition (21) has been updated to use $\boldsymbol{v}$. And even though $\boldsymbol{v}$ is not useful by itself and can be discarded after the solve, its contribution allows for a more accurate bubble *inertia aware* estimation of $\boldsymbol{u}$ and P, as opposed to the bubble *inertia unaware* formulation (16). The finite difference in equation (20) is computed with respect to

$$\tilde{\boldsymbol{v}}^{k+1} = \boldsymbol{v}^{k+1} + \theta\frac{\nabla P^k \Delta t}{\rho_b}, \tag{22}$$

which is $\boldsymbol{v}^{k+1}$ with the buoyancy update removed, as we are re-solving for it. We have also introduced a *compliance coefficient* $\theta \in [0, 1]$ as a means of adjusting the fraction of bubble inertia, to take into account in (20), and consequently the fraction of bubble buoyancy to re-solve for in (22).

As before, to solve the system of equations (19), (20), and (21) numerically we build its discrete representation as

$$\frac{1}{\Delta t}\boldsymbol{D}_\star\left(\boldsymbol{u} - \boldsymbol{u}^k\right) = \boldsymbol{f}_\star - \boldsymbol{G}P, \tag{23}$$

$$\frac{1}{\Delta t}\boldsymbol{D}_b\left(\boldsymbol{v} - \tilde{\boldsymbol{v}}^{k+1}\right) = -\theta \boldsymbol{G}P, \tag{24}$$

$$\boldsymbol{G}^T\left(\boldsymbol{\Phi}_w \boldsymbol{u} + \boldsymbol{\Phi}_b \boldsymbol{v}\right) = \boldsymbol{0}. \tag{25}$$

Applying $\boldsymbol{G}^T\boldsymbol{\Phi}_w\boldsymbol{D}_\star^{-1}\Delta t$ to (23) and $\boldsymbol{G}^T\boldsymbol{\Phi}_b\boldsymbol{D}_b^{-1}\Delta t$ to (24), and substituting $\boldsymbol{G}^T\boldsymbol{\Phi}_w\boldsymbol{u}$ and $\boldsymbol{G}^T\boldsymbol{\Phi}_b\boldsymbol{v}$ into (25) yields

$$\begin{aligned}\Delta t \boldsymbol{G}^T\left(\boldsymbol{\Phi}_w\boldsymbol{D}_\star^{-1} + \theta\boldsymbol{\Phi}_b\boldsymbol{D}_b^{-1}\right)\boldsymbol{G}P \;=\; \\ \boldsymbol{G}^T\left(\boldsymbol{\Phi}_w\left(\boldsymbol{u}^k + \Delta t\boldsymbol{D}_\star^{-1}\boldsymbol{f}_\star\right) + \boldsymbol{\Phi}_b\tilde{\boldsymbol{v}}^{k+1}\right),\end{aligned} \tag{26}$$

which differs from equation (18) by introducing the bubble *inertia correction* term $\theta\boldsymbol{\Phi}_b\boldsymbol{D}_b^{-1}$ to the diagonal scaling of the Laplacian. This correction together with (22) facilitates better overall convergence

of the coupling scheme and allows us to use fewer Newton iterations in practice, especially with larger bubble fractions ϕ_b.

3.5 Discussion

Figure 5 shows bubbles coupled with an Eulerian fluid simulated using our method. The bubble radii are varied from 0.5 mm to 0.5 cm using the distribution function presented later in Section 5. The fluid is represented sparsely within a narrow band of the bubble particles, and a hydrostatic pressure boundary condition is enforced on the outside. A kinematic animated toy also influences the pressure solve through a Neumann boundary condition. Though not seen in the static image, the bigger bubbles tend to be primarily influenced by the buoyancy force and hence rise faster compared to smaller bubbles, whose motion is primarily dominated by the fluid drag force. This natural breakup between the bubble scales is the result of our use of a physically based drag force model (5). Drag coefficient $\chi_b = 1$ is a great default value which we have used for all of the examples in this paper. Deviating from it may help address artistic notes, but we have found adjusting bubble size distribution to be a more intuitive control.

The compliance fraction $\theta = 0$ corresponds to inertia unaware pressure projection (18), where bubble velocities are "locked". On the other hand, the inertia aware regime $\theta = 1$ allows the fluid to push the bubbles around. Both regimes perform on par when the content of the bubbles is not high: with bubble fractions $\phi_b < 0.1$ we observed minimal visual difference between the two. However, as bubble fractions ϕ_b get closer to 1, the inertia unaware approach tends to experience convergence issues and occasionally leads to instabilities as shown in Figure 6, while the inertia aware results remain stable. Figure 7 also demonstrates velocity convergence plots for a single bubble simulated with both techniques.

The inertia aware approach accounts for bubble inertia but still fails to fully resolve for the drag between $\boldsymbol{v}$ and $\boldsymbol{u}$, as $\boldsymbol{u}$ is dragged only with respect to $\boldsymbol{v}^k$, and $\boldsymbol{v}$ is not dragged at all. Incorporating fully implicit treatment of drag into the system of equations (19), (20), and (21) can be done similar to [Fei et al. 2018]; we leave it for future work, as it would complicate the implementation considerably. We have found using $\theta = 0.5$ to be a reliable compromise between

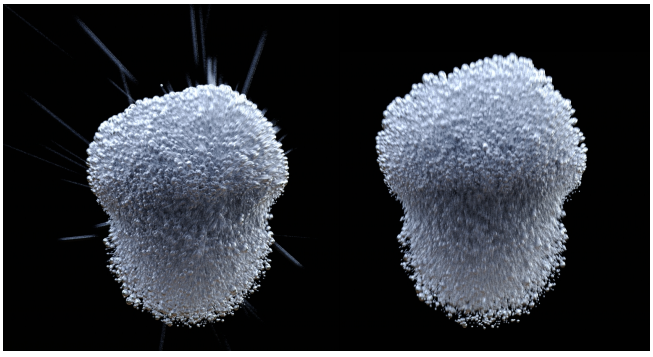

Fig. 6. A bubble inertia unaware simulation (left) experiences convergence issues and instabilities when the content of air is high ($\phi_b = 0.7$). These issues are manifest as streaky artifacts from fast moving particles. In contrast, an inertia aware simulation (right) remains stable. ©Wētā FX.

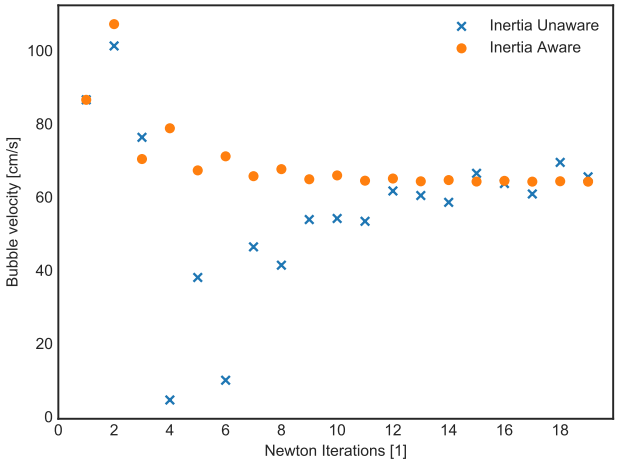

Fig. 7. The inertia aware ($\theta = 1$) approach tends to exhibit better convergence properties than the inertia unaware ($\theta = 0$) one. Here a bubble was placed at the center of an Eulerian grid voxel of size 6.25 mm, and its radius was chosen such that the bubble fractions at the adjacent faces would be $\phi_b = 0.9$. The initial velocity was set to 100 cm/s upwards. The simulation was run for one timestep of size 1/24 s with the bubble velocity recorded for each Newton iteration. ©Wētā FX.

"locked" and "freely sliding" bubbles, which consistently delivers stable results.

The equations of motion (6), (7), and (3) are not well-defined when bubble fractions ϕ_b become equal to 1. Practically, nothing prevents a user from packing a large number bubbles into a single voxel and even exceeding 1. And if not on purpose, this can happen sporadically as the bubbles drift over the course of a simulation. To deal with this inconvenience, we introduce a clamp $\phi_b^{\max}$ on what the maximum ϕ_b is allowed to be. Typically, we would have it set to a value between 0.3 and 0.7, depending on the maximum air saturation we expect a simulation to have. We then compute

$$s = \max\left(\frac{\phi_b}{\phi_b^{\max}}, 1\right),$$

which signifies the excessive relative amount of air on each face. ϕ_b is then replaced by $\phi_b/s \leq \phi_b^{\max}$. Additionally, s is interpolated to the particles, and their volumes are divided by it for the duration of the timestep to ensure momentum-conserving buoyancy force exchange.

4 GUIDED BUBBLES

It is often advantageous to simulate bubbles as a secondary simulation to enhance a pre-cached bulk water simulation. Running bubble simulations without having to simultaneously simulate a very expensive high-resolution bulk water reduces iteration time and gives artists more granular control. This workflow is especially important in production scenarios where the look of the bulk water has already been approved and the artist would like to add bubble effects without affecting the bulk water. However, simply driving bubble dynamics by drag and buoyancy forces with respect to the bulk fluid *unidirectionally* results in a disconnected look that then requires significant artist tuning to improve. To solve this problem

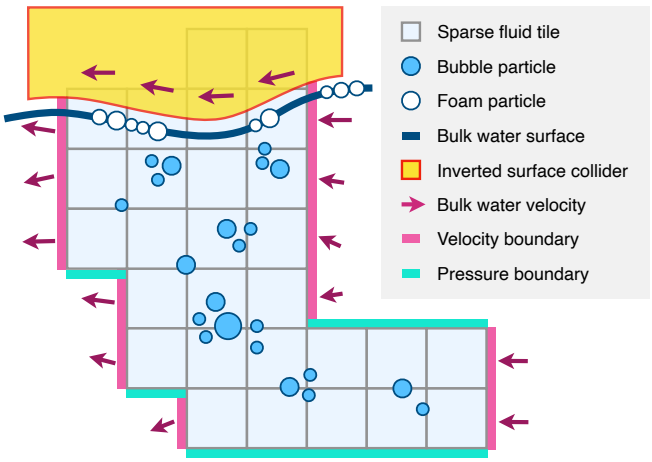

Fig. 8. Schematic representation of our guided bubble simulation approach shown in 2D for clarity. Bubbles are coupled with a fluid sparsely represented in their vicinity. The fluid is influenced by an existing bulk water cache through velocity and pressure boundary conditions. Additionally, the bulk water surface is inverted and used as a collider. Bubbles that reach the bulk water surface are either discarded or turned into foam. ©Wētā FX.

we introduce our *guided* bubble simulation method. This approach provides high-fidelity collective bubble effects with the benefits of running as a secondary simulation.

We achieve a natural looking bubble-water interaction by creating a sparse, two-way coupled, volumetric fluid simulation around bubble regions that are guided by an existing bulk water motion through boundary conditions; see Figure 8. This method effectively performs a local re-simulation of fluid in the vicinity of the bubbles. We use a tiled approach to allocate the fluid volume around bubble particles at each timestep, where a tile of fluid, typically $8 \times 8 \times 8$ voxels in size, is created if there are bubbles inside of it. Additional layers of tiles may be added for extra padding. We have found that setting the voxel size of the sparse fluid volume to be 1-2x the voxel size of the original bulk water simulation works well.

We set the boundary conditions on the borders of the sparse volume based on an existing bulk water cache that surrounds the sparse volume. Therefore, as we solve for the sparse fluid and bubble velocities as previously described in Section 3, these boundary conditions effectively guide the simulation and connect it to the surrounding bulk water.

Enforcing a velocity boundary condition on the boundary faces of the sparse fluid volume using pre-cached bulk water velocity provides natural-looking fluid currents. However using pure Neumann boundary conditions in the pressure solve is generally ill-defined. And while the null modes can be projected out gracefully, see [Bridson 2015], discretization errors may still lead to undesirable compression or expansion effects. Enforcing a pressure boundary condition does not have that problem, but it struggles to faithfully represent the motion of the bulk water. We thus choose to enforce the velocity boundary condition on the vertical boundary faces of the sparse fluid domain, and the pressure boundary condition in the voxels that are immediately in contact with the horizontal faces. We also use the bulk fluid surface as a collider, to guarantee no normal motion with respect to it.

Figure 9 provides an example simulation using our guided approach and compares it to the unidirectional one. The benefits of the guided approach are clearly apparent, with the fine detail and vorticity lost in the absence of proper coupling.

5 BUBBLE EMISSION

We have described how to simulate high-fidelity collective bubble effects using our guided, two-way coupled approach. However, an aspect that is often overlooked in simulation literature is emission: when should we emit bubbles, and how should they be initialized? Because we focus on the goal of providing an end-to-end simulation method that produces believable results and does not require significant parameter tuning, we regard the importance of emission similarly to the simulation method itself. We take a principled approach to examine the properties of the bulk simulation, which we then use to perform bubble emission. This provides the natural, connected look we seek and works well across many scenarios. The combination of emission and simulation ultimately determines the intricate distributions, shapes, and motion that are crucial to the final look of bubbles and foam.

We will first describe our physically based "aeration" metric, which is based on [Gualtieri et al. 2008] and introduced here to the graphics community. Aeration determines where bubbles are emitted. We then describe how to determine the quantity of bubbles to create and what their radii should be. With a plausibly varied distribution of bubble radii and drag/buoyancy balance having a different effect across scales, we achieve the interesting behaviors discussed in Section 3.5.

5.1 Aeration

[Gualtieri et al. 2008] argues that entrainment of air in a liquid happens when the local shear stress exceeds that of surface tension. Consequently, we define the dimensionless *aeration* metric

$$\mathcal{A} = \frac{2Ha\rho_w \|[\boldsymbol{u}] \otimes [\boldsymbol{u}]\|_F}{\pi\gamma}, \tag{27}$$

by dividing instantaneous Reynolds stress $\rho_w \|[\boldsymbol{u}] \otimes [\boldsymbol{u}]\|_F$ by surface tension pressure $\pi\gamma/2Ha$. Here γ, H, a, and $[\boldsymbol{u}]$ are the surface

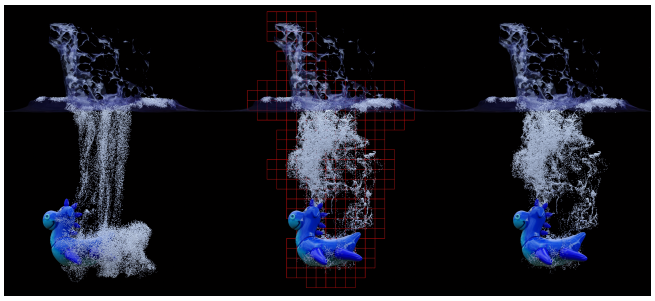

Fig. 9. Using a pre-cached bulk water motion to drive a bubble simulation unidirectionally (left) has significantly less motion detail and vorticity than our guided technique (right), which re-simulates fluid on a sparsely allocated set of tiles (middle) local to and two-way coupled with the bubbles. Coupling also yields the characteristic collective effect where denser groups of bubbles rise faster than they otherwise would individually, thus preventing them from getting "stuck" at the bottom. ©Wētā FX.

tension coefficient of water, water surface mean curvature, characteristic area of deformation, and local velocity fluctuation, respectively. $\otimes$ and $\|\cdot\|_F$ denote the outer product and matrix Frobenius norm. With this formulation, an aeration value $\mathcal{A} \gtrsim 1$ indicates an onset of air entrainment.

We calculate velocity fluctuation as the distance from the *averaged* velocity value $\overline{\boldsymbol{u}}$

$$[\boldsymbol{u}] = \boldsymbol{u} - \overline{\boldsymbol{u}}, \tag{28}$$

where $\overline{\boldsymbol{u}}$ is obtained by applying a low-pass convolution filter g in space and time to the bulk fluid velocity field $\boldsymbol{u}$

$$\overline{\boldsymbol{u}}(\boldsymbol{x}, t) = \int_{\mathbb{R}^3} \int_{-\infty}^{\infty} \boldsymbol{u}(\boldsymbol{y}, \tau) g(\boldsymbol{x} - \boldsymbol{y}, t - \tau) d\tau d\boldsymbol{y}. \tag{29}$$

We approximate equation (29) by using a simple box filter of size Δx in space and Δt in time. The averaged velocity then gets computed at the center of a voxel $\boldsymbol{i}$ as

$$\overline{\boldsymbol{u}}_{\boldsymbol{i}}^n = \frac{1}{4} \sum_{m=n-1}^{n} \sum_{(\boldsymbol{j},\alpha)\in\Omega(\boldsymbol{i})} u_{\boldsymbol{j},\alpha}^m \boldsymbol{e}_\alpha. \tag{30}$$

where superscripts n and m denote evaluation at the corresponding timestep, and $\Omega(\boldsymbol{i})$ is the set of 6 faces $(\boldsymbol{j}, \alpha)$ of a voxel $\boldsymbol{i}$. The velocity fluctuation then becomes

$$[\boldsymbol{u}]_{\boldsymbol{i}}^n = \frac{1}{4} \sum_{(\boldsymbol{j},\alpha)\in\Omega(\boldsymbol{i})} \left(u_{\boldsymbol{j},\alpha}^n - u_{\boldsymbol{j},\alpha}^{n-1} \right) \boldsymbol{e}_\alpha. \tag{31}$$

Mean curvature H can be computed directly from the distance field $\Phi(\boldsymbol{x}, t)$ associated with the fluid surface as

$$H(\boldsymbol{x}, t) = -\frac{1}{2} \nabla \cdot \left(\frac{\nabla \Phi(\boldsymbol{x}, t)}{\|\nabla \Phi(\boldsymbol{x}, t)\|} \right), \tag{32}$$

and we set the characteristic surface area $a = (\Delta x)^2$. Combining all of the above, the aeration $\mathcal{A}_{\boldsymbol{i}}^n$ can readily be obtained. We precompute volumetric aeration during bulk fluid simulation and sample it later for bubble emission.

Larger values of $\mathcal{A}$ correspond to more air being entrained. However, this does not give a quantitative estimate of the actual air volume. We thus define the *target volume fraction* to be filled with bubbles via a remap of $\mathcal{A}$ from the user-provided range $[\mathcal{A}_{\min}, \mathcal{A}_{\max}]$ to $[0, \phi_b^{\max}]$

$$\phi_b^{\text{target}} = \phi_b^{\max} \frac{\mathcal{A} - \mathcal{A}_{\min}}{\mathcal{A}_{\max} - \mathcal{A}_{\min}}, \tag{33}$$

so $(\Delta x)^3 \phi_b^{\text{target}}$ represents how much air is to be entrained in a single voxel. Values $\mathcal{A}_{\min}$ and $\mathcal{A}_{\max}$ represent the main artistic controls for adjusting the target emission volume and are typically set in the $[1, 100]$ range.

Equation (27) reduces the typical artist workflow from combining multiple emission heuristics such as velocity, curvature, and vorticity, into a single, physically meaningful measure. Note however that any single heuristic based only on the liquid flow will always be insufficient to realistically represent real air entrainment; an example of this is whitecap formation on an ocean surface, as in addition to fluid parameters it is also influenced by the local wind speed.

5.2 Bubble random distribution function

The aeration heuristic indicates where bubbles should be created, but not what sizes they should be. The process of entrainment is complex: large air regions quickly break down into smaller bubbles, and the final bubble sizes are dependent on the local fluid properties, such as surface tension, surfactant concentration, and shear stress. We thus define our bubble size distribution based on the quantitative analysis done by [Deike et al. 2016]. They found that bubble sizes after a breaking wave can be approximated by an inverse cubic distribution. As such, we define the *non-normalized* bubble size probability density function $Q(r) = 1/r^3$ as the measure of the likelihood of a bubble with radius r to form. We integrate it analytically given a radius range $r \in [r_{\min}, r_{\max}]$ to obtain the cumulative distribution function

$$\mathcal{R}(r) = \frac{r_{\max}^2}{r^2} \frac{r^2 - r_{\min}^2}{r_{\max}^2 - r_{\min}^2}. \tag{34}$$

This function is monotonic and has a unique inverse $\mathcal{R}^{-1}$. Thus, a random radius can be obtained as $r_X = \mathcal{R}^{-1}(X)$ from a uniformly sampled variable $X \in [0, 1]$. In our examples we have bubbles ranging from one to a few millimeters in size.

The emission algorithm proceeds as follows. Since entrainment can only happen near the surface, we iterate over all voxels and discard the ones that are more than $2\Delta x$ away from the water surface. For each of the remaining voxels, we compute the residual volume $(\phi_b^{\text{target}} - \phi_b)(\Delta x)^3$. We then repeatedly sample $\mathcal{R}$ and create new bubbles until the total target volume is reached.

If new bubble positions are chosen uniformly within a voxel, clear boundaries are visible between voxels with different aeration values. We resolve this by sampling the uniform distribution within a voxel multiple times and adding the off-center displacements together. Assuming independence of samples, we effectively replace a sharp indicator function of a voxel by a linear or quadratic B-spline, corresponding to adding 2 and 3 samples respectively. Note, that this approach inevitably creates bubbles outside of their intended voxel, but we have not found this to be visually distracting. Any bubble that would get emitted outside of the liquid is deleted.

6 FOAM SIMULATION

With a robust underwater bubble emission and simulation method that can produce convincing aerated water effects, we will now address the behavior of foam; that is, bubbles that have reached the water surface. Rather than formulating a complex monolithic foam model that captures many types of foam, we focus our method on "wet" foam which is typical of the water scenarios we target. The assumption of wet foam allows us to take a principled, yet simplified approach by modeling the foam as a viscous fluid. Our foam is constrained to the water surface using a method we introduce called *manifold advection*. These simplifications reduce the complexity of our technique, allowing it to efficiently scale to large-scale scenarios while maintaining high-fidelity foam shapes and motion that are seamlessly integrated with the water surface.

6.1 Physics

The Bingham constitutive model is a common modeling choice for foam ([Kroezen et al. 1988; Weaire and Hutzler 2001]) and separates the rheology of foam into two distinct regimes. Under small perturbations the foam exhibits elastic behavior. It is characterized by the foam bubbles forming a lattice structure that locks each individual bubble in place. As the internal elastic stress reaches a yield stress value σ_y, it is no longer possible for the structure to hold its shape, and the foam flows, exhibiting plastic behavior.

Strictly speaking, foam flow is not continuous. As the critical stress is reached, some bubbles will—almost instantaneously—reorder themselves to a new topology with lower potential energy. A single topology change event tends to trigger other bubbles to reorder in a cascade, which eventually causes the material tension to fall below the point of yielding. Accurately modeling these events is challenging as they are affected by a myriad of microscopic parameters such as bubble size, surfactant concentration, and liquid-to-air ratio. What is more, the bubbles themselves are often unstable and can burst at any moment, making accurate modeling even more challenging.

One important metric of foam is that of *wetness*, which is measured as the ratio between liquid and gas in a sufficiently large local volume and may vary spatially. For example, when pouring a glass of carbonated beverage a thick layer of foam forms. Gravity will cause the liquid to be drained from the top layers, causing the foam to be relatively more wet at the bottom. Wetness has an important effect on the flowing properties of foam as it affects the yield stress σ_y. A quantitative value of this relationship will depend on the particular foam in question, but generally σ_y decreases as wetness increases.

Wetness affects the shapes of individual bubbles as demonstrated in [Dunne et al. 2017], Figure 2. When foam is dry the lack of liquid between bubbles tends to deform the bubbles into patterns resembling a Voronoi diagram. A higher liquid content allows the bubbles to stay more spherical. It is not possible to directly determine the liquid-to-air ratio from the shapes of the bubbles in general; as even if they are nearly spherical, if large differences in bubble sizes are present, dry foams can still be formed by packing tiny bubbles in between large ones.

From the close-up sea foam reference in Figure 10 we glean two things: a) the bubbles look roughly spherical and b) they are similar in size. We will take these empirical observations and claim that the foams we are interested in modeling are *wet*. As a consequence, we argue that we can take $\sigma_y = 0$ from here on, and hence ignore the elasticity of foam, to model it simply as a viscous fluid. While this approximation cannot reproduce certain features like thick stacking foams, such as those seen in carbonated beverages or during certain sea conditions, we show that it is sufficient to model a large range of common whitewater scenarios in our examples. This simplification also affords us the ability to scale to large production-level scenarios which would be difficult with a more complex model.

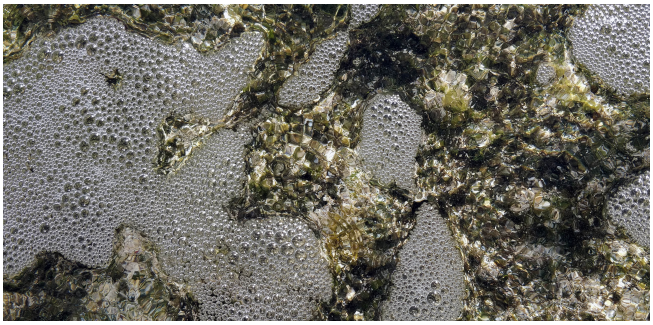

Fig. 10. A reference image of ocean foam in Hawaii. ©Wētā FX.

6.2 Foam as a viscous fluid

We approximate the motion of foam as that of a Newtonian fluid

$$\rho \frac{D\boldsymbol{v}}{Dt} = -\nabla P + \mu \nabla^2 \boldsymbol{v} + \boldsymbol{g}, \tag{35}$$

where $\frac{D}{Dt}$ is the material derivative and $\boldsymbol{v}$, ρ, P, and μ are foam velocity, density, pressure, and viscosity, respectively. Additional forces that may also be present on the right hand side of the equation (35), such as cohesion, which we will introduce below.

Pressure arises from individual bubble-bubble interactions, and it would make sense to use as an equation of state that of an ideal gas with $P = \kappa\rho$ for some stiffness coefficient κ. However, the strong elastic forces that would keep dry foam together are not present, due to the assumption of the foam being wet. When pairs of bubbles are pulled apart, the large reservoir of water that is available by being at the fluid surface ensures that individual bubbles keep their shape, and the foam freely separates rather than stretches. As a result, low pressure regions are unlikely to form in such foams. We remove these zones from the equation of state by clamping pressure in areas where density is lower than the rest density ρ_0, and define

$$P = \max(\kappa(\rho - \rho_0), 0). \tag{36}$$

Viscosity is the emergent behavior from the bubble topology change events mentioned earlier, and its value will increase with the number of topology change events in any particular foam. It is therefore dependent on the average bubble diameter of the foam: a foam with smaller bubbles will exhibit higher apparent viscosity than that of a foam with comparatively larger bubble sizes.

6.3 Smoothed particle hydrodynamics

We represent foam as a set of points $\mathcal{P}$. Each point $p \in \mathcal{P}$ stores position $\boldsymbol{x}_p$, velocity $\boldsymbol{v}_p$, and radius r_p. In the derivations below we also use volume V_p and mass m_p, but those are calculated from the radius assuming particles are perfect spheres and have uniform density ρ_0. The equation of motion (35) reduces to

$$m_p \frac{\mathrm{d}\boldsymbol{v}_p}{\mathrm{d}t} = \boldsymbol{F}_p, \tag{37}$$

where $\boldsymbol{F}_p$ represents the collective effect of all forces acting upon a particle p, such as pressure and viscosity. Although the air inside of bubbles can diffuse through the bubble membrane, we choose not to model this behavior to reduce complexity, and we leave this type of interaction for future work. As a result, the mass and radius of each particle are considered constant.

SPH creates continuous fields $A(\boldsymbol{x})$ from discrete particle values A_p, via an interpolation kernel $W(\boldsymbol{x}, h)$ with finite support radius h

$$A(\boldsymbol{x}) = \sum_{q \in \mathcal{P}} V_q A_q W(\boldsymbol{x} - \boldsymbol{x}_q, h_q).$$

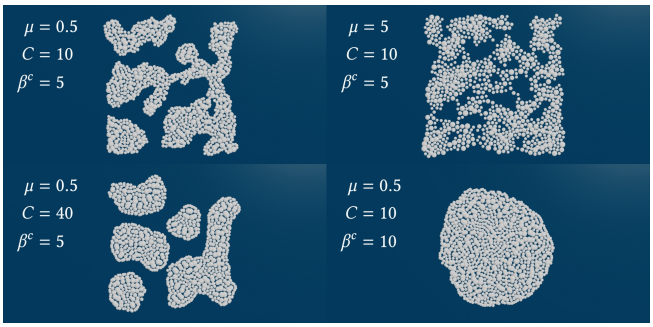

Fig. 11. A wedge of different SPH parameters. Increasing cohesion C (bottom left) leads to particles clumping together faster, while increasing β^c (bottom right) results in larger clumps. Higher viscosity values (top right) dampen the relative motion between particles, which is characteristic of foams made out of smaller bubbles. ©Wētā FX.

Similarly, gradients are calculated by

$$\nabla A(\boldsymbol{x}) = \sum_{q\in\mathcal{P}} V_q A_q \nabla W(\boldsymbol{x} - \boldsymbol{x}_q, h_q).$$

Typically, $W(\boldsymbol{x}, h)$ is symmetric and only depends on the magnitude of $\boldsymbol{x}$. Note that the support varies per particle, and is calculated as a multiplier of the radius $h_p = \beta r_p$, where β is a user-defined constant. To ensure symmetric calculations between points p and q we average the support radius as

$$h_{pq} = \frac{h_p + h_q}{2}, \tag{38}$$

and define $W_{pq} = W(\boldsymbol{x}_p - \boldsymbol{x}_q, h_{pq})$.

6.4 Forces

We will now discuss forces acting on a foam particle. As in a typical SPH formalism they will be presented as per-particle accelerations.

Pressure. Acceleration due to pressure $\boldsymbol{a}_p^P$ is calculated using the numerically stable formulation from [Monaghan 1992]

$$\boldsymbol{a}_p^P = -\sum_{q\in\mathcal{P}} m_q \Big(\frac{P_p}{\rho_p^2} + \frac{P_q}{\rho_q^2}\Big)\nabla W_{pq}, \tag{39}$$

where the *smoothed* density ρ_p is computed as

$$\rho_p = \sum_{q\in\mathcal{P}} m_q W_{pq} \tag{40}$$

and pressure is calculated using equation (36). The clamp in equation (36) is also a common way to improve stability of SPH pressure forces in low-density regions [Akinci et al. 2013]. We use $\kappa = 0.5\ \mathrm{m}^2\mathrm{s}^{-2}$ in all our simulations.

Viscosity. We follow the suggestion from [Monaghan 1992] and define viscous acceleration as

$$\boldsymbol{a}_p^\mu = \begin{cases} -\sum_{q\in\mathcal{P}} m_q \Pi_{pq} \nabla W_{pq}, & \text{if } \boldsymbol{v}_{pq}\cdot\boldsymbol{x}_{pq} < 0, \\ 0, & \text{if } \boldsymbol{v}_{pq}\cdot\boldsymbol{x}_{pq} \geq 0, \end{cases} \tag{41}$$

where

$$\Pi_{pq} = \mu \frac{2h_{pq}}{\rho_p + \rho_q} \frac{\boldsymbol{v}_{pq}\cdot\boldsymbol{x}_{pq}}{\|\boldsymbol{x}_{pq}\|^2 + \xi^2}, \tag{42}$$

and we define $\boldsymbol{w}_{pq} = \boldsymbol{w}_p - \boldsymbol{w}_q$ for an arbitrary vector $\boldsymbol{w}$. Parameter ξ ensures the acceleration stays bounded as $\boldsymbol{x}_{pq} \to \boldsymbol{0}$, and is typically set to $0.1h_{pq}$. The condition $\boldsymbol{v}_{pq}\cdot\boldsymbol{x}_{pq} \geq 0$ is the SPH equivalent of $\nabla\cdot\boldsymbol{v} \geq 0$. Equation (41) is an artifical viscosity formulation, and the parameter μ relates to the speed of sound of the material. All of our results, except for the examples shown in Figure 11, have been run using $\mu = 0.05$ m/s.

Cohesion. Cohesion helps capture bubble-bubble attraction forces that are often seen for bubbles on a liquid surface. This phenomenon is sometimes called the "cheerios effect" [Vella and Mahadevan 2005] and is caused by the liquid meniscus formed around each bubble having a surface that is not flat. The fluid buoyancy will tend to push bubbles up along the slanted surface, causing them to cluster. Although it would be tempting to use SPH particle geometry to estimate local surface deformation, we leave it for future work and use a simple cohesion force instead.

In the SPH literature there are plenty of cohesion-like forces to choose from, such as [Akinci et al. 2013; He et al. 2014]. We use a formulation inspired by [Becker and Teschner 2007] and define the acceleration due to cohesion as

$$\boldsymbol{a}_p^c = -C \sum_{q\in\mathcal{P}} V_p \frac{\boldsymbol{D}_{pq}}{h_{pq}^c} W(\boldsymbol{x}_p - \boldsymbol{x}_q, h_{pq}^c), \tag{43}$$

with two differences. Firstly, our formulation uses the effective distance $\boldsymbol{D}_{pq} = \boldsymbol{x}_{pq} - (r_p + r_q)\frac{\boldsymbol{x}_{pq}}{\|\boldsymbol{x}_{pq}\|}$ instead of the distance between particle centers. This limits the compression caused by cohesion, and removes the need to increase pressure stiffness as cohesion stiffness increases. Secondly, we divide by the support radius, similar to [Akinci et al. 2013]. The cohesion support radius $h_p^c = \beta^c r_p$ for some user input β^c may be chosen differently from the support h_p used for pressure and viscosity computations. This is because we have found both the support radius and the cohesion strength C to be important artistic controls to addresss look requirements. The division by h_{pq}^c ensures that the potential energy stays bounded with changes in the support radius.

Fluid surface drag. Liquid velocity $\boldsymbol{u}$ has a significant impact on the dynamics of foam. For $\boldsymbol{u} \neq \boldsymbol{v}_p$ there will be a thin liquid layer with height d over which a viscous force would act to reduce the difference [Persson and Dahlberg 1994]. Instead of providing the characteristic height d and viscosity as user parameters, we express the interaction as a linear drag acceleration

$$\boldsymbol{a}^\Phi = \chi^\Phi\big(\boldsymbol{u}(\boldsymbol{x}_p) - \boldsymbol{v}_p\big), \tag{44}$$

for some user-defined drag rate χ^Φ with units [1/s]. Superscript Φ signifies that the force is applied along the surface of the fluid, which corresponds to the manifold constraint $\Phi = 0$ introduced later. Equation (44) is meaningful for any $\chi^\Phi \geq 0$, however in practice the range is more limited. If χ^Φ is "too high" the foam will tend to override any other material forces, causing a "stringy" look as particles are trapped in fluid vortices at the surface. Conversely, if the value is "too low" the foam will slide. We have found a useful range to be between 0.05 s^{-1} and 0.5 s^{-1}.

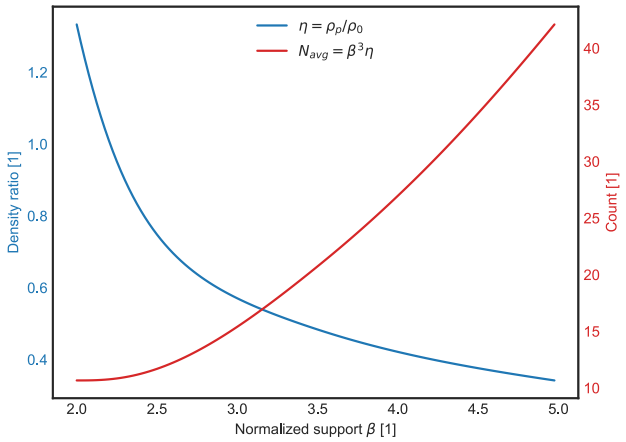

Fig. 12. Plot of density ratio $\eta = \rho_p/\rho_0$ and average particle count N_{avg} with support radius $h = \beta r$ for an infinite plane of perfectly packed particles of radius r and the kernel from [Monaghan 1992]. ©Wētā FX.

The total force per-particle $\boldsymbol{F}_p$ is obtained by adding the contributions of pressure, viscosity, cohesion, drag, and gravity

$$\boldsymbol{F}_p = m_p(\boldsymbol{a}_p^P + \boldsymbol{a}_p^\mu + \boldsymbol{a}_p^c + \boldsymbol{a}_p^\Phi + \boldsymbol{g}). \tag{45}$$

Examples of the different dynamic behaviors that can be achieved with the proposed SPH formulation can be seen in Figure 11.

6.5 SPH kernel considerations

We employ the cubic kernel suggested by [Monaghan 1992] defined as $W(\boldsymbol{x}, h) = h^{-3}\omega(2\|\boldsymbol{x}\|/h)$ and

$$\omega(q) = \frac{1}{\pi}\begin{cases} 1 - \frac{3}{2}q^2 + \frac{3}{4}q^3, & 0 \le q \le 1 \\ \frac{1}{4}(2-q)^3, & 1 \le q \le 2, \\ 0, & q > 2. \end{cases} \tag{46}$$

We note that with this formulation the support radius h represents the least upper bound of $\|\boldsymbol{x}\|$ for which $W(\boldsymbol{x}, h) > 0$.

We would like the particles to be optimally packed on the water surface to create connected foam regions. We thus analyze the following simplified scenario: an infinite plane of optimally packed spherical particles of the same radius r. For any particle p it is then possible to calculate the density ratio $\eta = \rho_p/\rho_0$ analytically. Assuming the particles are confined to the XY-plane, the position $\boldsymbol{x}^{lm}$ of a particle with index $(l, m) \in \mathbb{Z}^2$ in the lattice space is[1]

$$\boldsymbol{x}^{lm} = (2l + m \bmod 2, \sqrt{3}m, 0)r.$$

Each particle has an identical neighbourhood due to symmetry, and can be chosen to analyse the density. Let $\Gamma(l, m)$ be the set of particles that have $\|\boldsymbol{x}^{lm}\| < h$. The density ratio for the particle at the origin is then

$$\eta = \sum_{q \in \Gamma(l,m)} V_q W(\boldsymbol{x}^{lm}, h). \tag{47}$$

The ratio η is independent of density ρ_0 and only depends on $\beta = h/r$ and the choice of kernel. This can be seen by noting that $V_q/h^3 = 4\pi/3\beta^3$ and that the argument of ω is unitless. The configuration is at rest when $\eta = 1$; that is, no pressure forces are active. And since we would like the bubble particles to be perfectly packed on the fluid surface when at equilibrium we define

$$\rho_0 = \frac{\rho_f}{\eta(\beta)}, \tag{48}$$

given the user-provided foam density ρ_f and a value of β. We have consistently used $\rho_f = 1$ kg/m^3 and $\beta = 4$ in this work.

For the same perfectly packed scenario, equation (47) can also be used to calculate the expected number of particles inside of a sphere of radius h ([Dehnen and Aly 2012]) as

$$N_{\text{avg}} = \beta^3\eta. \tag{49}$$

Both η and N_{avg} as functions of β are shown in Figure 12.

6.6 Foam and water surface interaction

We are interested in the treatment of foam on top of a liquid interface, and in that pursuit it is essential that we have a robust and stable formulation that ensures that the foam accurately follows said interface. Gas bubbles by a liquid interface are at rest due to equilibrium between gravity, buoyancy, and surface tension forces [Vella and Mahadevan 2005]; but explicitly computing these forces and ensuring a stable balance is challenging. The scale of real-world bubbles is often orders of magnitudes smaller than the distance a fluid travels in a single timestep for typical graphics applications; and buoyancy and surface tension are especially difficult to resolve due to their reliance on an accurate representation of the fluid interface. We thus eliminate the normal component of all forces and replace those by a constraint, limiting the motion of foam particles to the surface.

Let $\Phi(\boldsymbol{x}, t)$ be a signed distance field that represents the liquid surface evolving with velocity $\boldsymbol{u}(\boldsymbol{x}, t)$. The normal $\boldsymbol{N}$ and tangent $\boldsymbol{T}$ components of a vector $\boldsymbol{w}$ at location $\boldsymbol{x}$ and time t with respect to Φ are defined as

$$\boldsymbol{N}(\boldsymbol{w}, \boldsymbol{x}, t) = \big(\nabla\Phi(\boldsymbol{x}, t) \cdot \boldsymbol{w}\big)\nabla\Phi(\boldsymbol{x}, t), \tag{50}$$

$$\boldsymbol{T}(\boldsymbol{w}, \boldsymbol{x}, t) = \boldsymbol{w} - \boldsymbol{N}(\boldsymbol{w}, \boldsymbol{x}, t). \tag{51}$$

The analysis hereafter is independent of which particular particle is being observed, and particle subscript p is omitted for brevity. We

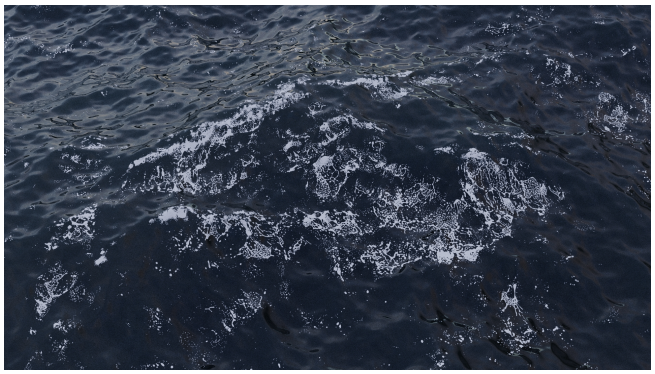

Fig. 13. 40m×40m region of foam simulated on top of a procedurally generated fluid surface using [Horvath 2015], showing that our foam method is not limited to close-up scenarios. This is possible due to the fast convergence of our manifold advection algorithm, removing the need to resolve the force equilibrium along the surface normal. ©Wētā FX.

[1] https://en.wikipedia.org/wiki/Close-packing_of_equal_spheres

want to enforce

$$\Phi(\boldsymbol{x}, t) = 0 \tag{52}$$

for every foam particle at all times. We call equation (52) the *manifold advection constraint*. Note that it explicitly limits us to only simulate foams that do not stack on the surface. In order to enforce the constraint we introduce a Lagrange multiplier λ and let

$$\boldsymbol{F_N} = \lambda \nabla \Phi(\boldsymbol{x}, t). \tag{53}$$

Physically, $\boldsymbol{F_N}$ is the force normal to the manifold that will ensure each bubble stays on the surface $\Phi = 0$. The tangential motion along the manifold is governed by all other forces $\boldsymbol{F}$ from equation (45).

Although $\boldsymbol{F}$ may not be in the tangent space of the manifold, due to $\boldsymbol{F_N}$, all acceleration in the normal direction that would violate equation (52) will be canceled out. For this reason we explicitly define the projection of the forces to the tangent space

$$\boldsymbol{F_T} = \boldsymbol{T}(\boldsymbol{F}, \boldsymbol{x}, t), \tag{54}$$

and the full equations of motion for foam can be written as

$$\begin{aligned} m\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} &= \boldsymbol{F_N} + \boldsymbol{F_T}, \\ \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} &= \boldsymbol{v}, \\ \Phi(\boldsymbol{x}, t) &= 0. \end{aligned} \tag{55}$$

6.7 Time integration

We will integrate equation (55) for each particle individually, by separating the tangential force term

$$\hat{\boldsymbol{v}}^n = \boldsymbol{v}^n + \frac{\Delta t}{m}\boldsymbol{F}_{\boldsymbol{T}}^n, \tag{56}$$

from advection subject to the manifold advection constraint

$$\begin{aligned} \boldsymbol{v}^{n+1} &= \hat{\boldsymbol{v}}^n + \frac{\Delta t}{m}\boldsymbol{F}_{\boldsymbol{N}}^n, \\ \boldsymbol{x}^{n+1} &= \boldsymbol{x}^n + \Delta t \boldsymbol{v}^{n+1}, \\ \Phi(\boldsymbol{x}^{n+1}, t^{n+1}) &= 0. \end{aligned} \tag{57}$$

We add a superscript n to signify evaluation at time t^n. The constraint in system (57) is taken with respect to the manifold at time t^{n+1} to ensure the particle is on the surface at the end of the simulation step.

All forces $\boldsymbol{F}$ are computed explicitly from equations (54) and (45), so calculating $\hat{\boldsymbol{v}}^n$is trivial. However, [Morgenroth et al. 2020] warn of potential dampening effects resulting from operator splitting the velocity update into force application and re-projection to the tangent space of the manifold. This can easily be seen by imagining a particle traveling on a trajectory constrained to a sphere, and gradually losing all its velocity as it traces a $\frac{\pi}{2}$ arc. [Morgenroth et al. 2020] argue that to mitigate this effect the velocity *magnitude* in the absence of external forces should be preserved, which implies rotating the velocity vector instead of projecting it. We thus introduce an additional velocity correction term

$$\Delta \boldsymbol{v}_R^n = \left(\frac{\|\boldsymbol{T}^{n-1}(\hat{\boldsymbol{v}}^{n-1})\|}{\|\boldsymbol{T}^n(\boldsymbol{v}^n)\|} - 1 \right) \boldsymbol{T}^n(\boldsymbol{v}^n),$$

and instead of (56) compute

$$\hat{\boldsymbol{v}}^n = \boldsymbol{v}^n + \frac{\Delta t}{m}\boldsymbol{F}_{\boldsymbol{T}}^n + \Delta \boldsymbol{v}_R^n, \tag{58}$$

to compensate for velocity magnitude loss due to re-projection. Here we have introduced $\boldsymbol{T}^n(\boldsymbol{w}) = \boldsymbol{T}(\boldsymbol{w}, \boldsymbol{x}^n, t^n)$, for an arbitrary vector $\boldsymbol{w}$. One can indeed confirm that in the absence of forces $\boldsymbol{F}_{\boldsymbol{T}}^n$ tangential velocity magnitude is preserved, that is $\|\boldsymbol{T}^n(\hat{\boldsymbol{v}}^n)\| = \|\boldsymbol{T}^{n-1}(\hat{\boldsymbol{v}}^{n-1})\|$. We store the value $v_{\boldsymbol{T}}^n = \|\boldsymbol{T}^n(\hat{\boldsymbol{v}}^n)\|$ to be used during velocity correction on next timestep.

Next we need to integrate the system (57). We do so by using a standard Newton-Raphson solver. Although not strictly necessary, the solver converges significantly quicker if only the tangential part of the velocity $\boldsymbol{T}^n(\hat{\boldsymbol{v}}^n)$ from equation (58) is used, which constitutes the projection to the manifold tangent space discussed above. It also has a negligible effect on the final solution, as $\boldsymbol{F_N}$ will remove normal motion that does not conform to equation (52) anyway. Additionally, we replace λ by a new unknown $\alpha = m\lambda/\Delta t^2$ to simplify the notations. The unknown position as a function of α becomes

$$\hat{\boldsymbol{x}}(\alpha) = \boldsymbol{x}^n + \boldsymbol{T}^n(\hat{\boldsymbol{v}}^n)\Delta t + \alpha \nabla \Phi(\boldsymbol{x}^n, t^n), \tag{59}$$

and needs to satisfy $\Phi(\hat{\boldsymbol{x}}(\alpha), t^{n+1}) = 0$. The Newton iteration can then be expressed as

$$\alpha^{k+1} = \alpha^k - \frac{\Phi(\hat{\boldsymbol{x}}(\alpha^k), t^{n+1})}{\alpha^k \nabla \Phi(\hat{\boldsymbol{x}}(\alpha^k), t^{n+1}) \cdot \nabla \Phi(\boldsymbol{x}^n, t^n)}. \tag{60}$$

A natural initial guess is given by $\alpha^0 = \Delta t \|\boldsymbol{N}(\boldsymbol{u}(\boldsymbol{x}^n, t^n), \boldsymbol{x}^n, t^n)\|$ which is the distance the fluid surface would travel in the normal direction over a simulation step.

We found that typically 2-3 iterations was enough to find positions that were within 0.1 mm of the fluid surface. However, due to potentially large topological changes that the surface may undergo,

ALGORITHM 1: Manifold advection

for step n **do**
- Require $\Phi(\boldsymbol{x}^n, t^n) = 0$, $v_{\boldsymbol{T}}^{n-1}$
- Calculate $\boldsymbol{F}^n$
- $\hat{\boldsymbol{v}}^n \leftarrow \boldsymbol{v}^n + \frac{\Delta t}{m}\boldsymbol{F}_{\boldsymbol{T}}^n + \Delta \boldsymbol{v}_R^n$
- $v_{\boldsymbol{T}}^n \leftarrow \|\boldsymbol{T}^n(\hat{\boldsymbol{v}}^n)\|$
- Solve $\Phi(\hat{\boldsymbol{x}}(\alpha), t^{n+1}) = 0$
- $\boldsymbol{v}^{n+1} \leftarrow (\boldsymbol{x}^{n+1} - \boldsymbol{x}^n)/\Delta t$

end

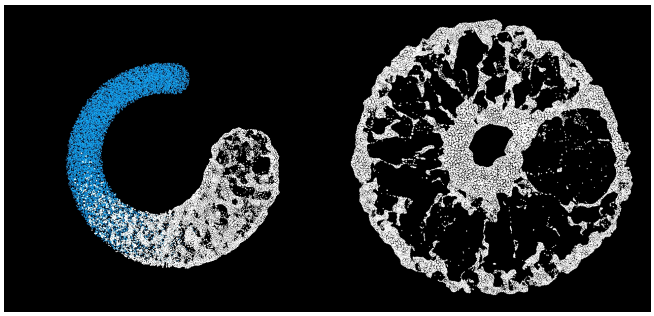

Fig. 14. The coupling presented in Section 3 is not limited to Eulerian fluids. Here bubbles (blue) are coupled with a FLIP fluid; and once they reach the surface, they transition into foam (white) and are simulated using SPH with manifold advection. Natural foam detail forms as the bubbles rise and impart significant inertia to the fluid. ©Wētā FX.

Fig. 15. **River.** A river flowing around sharp creases of a winding canyon creates a combination of calm and dynamic regions, including waterfalls and backdrafts. Our guided bubbles and foam simulation technique can capture the characteristic features of these behaviors. ©Wētā FX.

it is possible to find solutions $\hat{\boldsymbol{x}}$ such that the particles travel unnaturally large distances. The most common example is an overturning wave where spurious tunneling events may occur. In practice we limit the maximum correction the advection algorithm may perform using a threshold ϵ, and the final position is given by

$$\boldsymbol{x}^{n+1} = \begin{cases} \hat{\boldsymbol{x}}(\alpha^k), & \|\hat{\boldsymbol{x}}^k - \hat{\boldsymbol{x}}^0\| \leq \epsilon, \\ \hat{\boldsymbol{x}}(\alpha^0), & \|\hat{\boldsymbol{x}}^k - \hat{\boldsymbol{x}}^0\| > \epsilon. \end{cases} \tag{61}$$

We have had success using $\epsilon = 0.1$m for all of our simulations. The final velocity is then computed as $\boldsymbol{v}^{n+1} = (\boldsymbol{x}^{n+1} - \boldsymbol{x}^n)/\Delta t$. The full procedure is outlined in Algorithm 1. A particularly challenging example for our foam material model and manifold advection algorithm is shown in Figure 13. The large swells are accurately tracked, while not destroying the foam structure.

Due to the threshold condition in equation (61), and possibly a chance of not converging in equation (60), a foam particle may not end up on the surface. It is then important to run it through a re-classification algorithm, outlined in Section 7.1.

7 GUIDED BUBBLES AND WET FOAM

We have now presented two novel and separately useful techniques for simulating bubbles and foam, respectively. In this section we describe our method for unifying them into a single system.

Both bubbles and foam consist of particles representing gas enclosed by liquid. We represent these as two particle systems that simultaneously execute their respective simulation algorithms. We explicitly move particles between the two groups and transfer their momentum as they meet the criteria detailed below.

7.1 Transitions

Let $\mathcal{P}$ be the set of all particles, bubbles and foam included. At the end of timestep n we define

$$\begin{aligned} \mathcal{S} &= \{p : \|\Phi(\boldsymbol{x}_p^n, t^n)\| \leq r_p\} \\ \mathcal{C} &= \{p : \text{sign}(\Phi(\boldsymbol{x}_p^n, t^n)) \neq \text{sign}(\Phi(\boldsymbol{x}_p^{n-1}, t^{n-1})\} \\ \mathcal{U} &= \{p : \Phi(\boldsymbol{x}_p^n, t^n) < 0\}. \end{aligned} \tag{62}$$

$\mathcal{S}$ contains the particles that are within their own radius from the water surface, $\mathcal{C}$ represents the ones that have crossed the surface during the timestep, and $\mathcal{U}$ the ones that are contained inside of the water SDF. Internally, we represent $\mathcal{S}$, $\mathcal{C}$, and $\mathcal{U}$ as bitmasks stored per particle. We can then decide whether a particle p should be treated as part of foam group $\mathcal{F}$, or bubble group $\mathcal{B}$; or deleted using Algorithm 2.

Any particle moving from $\mathcal{B}$ to $\mathcal{F}$ is projected onto the surface Φ to ensure that the initial condition for the manifold constraint equation (52) is satisfied at the start of the next timestep. Generally speaking, the momentum of such a particle cannot be preserved through transition, since it is now constrained to the liquid surface. Thus, we specify how much momentum to conserve using a parameter $\zeta \in [0, 1]$

$$\boldsymbol{v}_{\text{after}}^n = (\zeta - 1)\|\boldsymbol{v}_{\text{before}}^n\| \frac{\boldsymbol{T}^n(\boldsymbol{v}_{\text{before}}^n)}{\|\boldsymbol{T}^n(\boldsymbol{v}_{\text{before}}^n)\|},$$

where subscripts 'before' and 'after' indicate the velocity of the particle before and after the transition, respectively. All of our simulations use $\zeta = 0.7$. An example of transitions performed with our algorithm can be seen in Figure 14.

7.2 Foam bursting

There are many processes that cause foam topology to change: liquid drainage, shear stresses, evaporation, and so on. Modeling those effects is outside the scope of this work. Instead, we store the age of foam particles during the simulation and let the user choose a probability distribution function that models bursting. Using a normal distribution defined by its mean and variance has proven to work well in practice.

As bubbles rise up to the surface and transition into foam, large SPH density ratios ρ_p/ρ_f may be observed. These have a tendency to lead to undesirably rapid expansions. We address this by pruning newly created foam particles using the following approach. Let $N_{\mathcal{P}}(\boldsymbol{x}, R)$ be the number of particles in $\mathcal{P}$ inside of a sphere with center $\boldsymbol{x}$ and radius R. For a newly transitioned particle $p \in \mathcal{F}$, we remove it if

$$\frac{N_{\text{avg}}}{N_{\mathcal{P}}(\boldsymbol{x}_p, h_p)} < X,$$

ALGORITHM 2: Particle group transitions

```
for p ∈ P do
    if p ∈ S or p ∈ C then
        if p ∈ B then
            Project x_p onto Φ
        end
        Move p to F
    else
        if p ∈ U then
            Move p to B
        else
            Delete p
        end
    end
end
```

Table 3. Summary of parameters with units and ranges of values used in our simulations.

Symbol	Units	Min	Max	Meaning
$\mathcal{A}_{\min}, \mathcal{A}_{\max}$	1	1	100	min/max aeration values
$r_{\min}, r_{\max}$	mm	0.5	5	min/max bubble radii
$\phi_b^{\max}$	1	0.3	0.7	max bubble fraction
χ_b	1	1	1	bubble drag coefficient
κ	$m^2 s^{-2}$	0.5	0.5	foam pressure stiffness
μ	m/s	0.05	5	foam viscosity coefficient
C	$m \cdot s^{-2}$	10	100	foam cohesion coefficient
β^c	1	5	12	foam cohesion radius
χ^Φ	s^{-1}	0.05	0.5	foam surface drag
$E(\tau)$	s	1.5	2	foam lifespan mean
$Var(\tau)$	s^2	0.5	0.5	foam lifespan variance

using a uniformly sampled variable $X \in [0, 1]$. This means we stochastically limit formation of new foam particles if the surface is already saturated and perfectly packed; see equation (49).

8 RESULTS

In addition to the simple tests that we covered in previous sections, we have also simulated a variety of larger scale examples that demonstrate the power of our method. For each of the examples we ran a guided bubbles and foam simulation on top of a pre-baked bulk water. We have come up with a diverse set of bulk fluid simulations which would stress test our whitewater system in different scenarios. We demonstrate lapping waves hitting a rocky beach scenario in Figure 1. A large submarine emerges from under water and creates a giant wake in Figure 2. Partially submerged spinning propellers induce characteristic flow patterns in Figure 3. And finally, Figure 15 shows a mountain river filled with waterfalls and backdrafts.

To dial in looks, we have found only a handful of parameters need tuning. In particular, the aeration input $[\mathcal{A}_{\min}, \mathcal{A}_{\max}]$ and bubble size $[r_{\min}, r_{\max}]$ ranges were necessary to adjust the underwater bubble distributions. Cohesion C, cohesion radius β^c, manifold drag χ^Φ, and particle lifespan helped achieve the desired foam patterns. A summary of all parameters with their respective units and recommended ranges is presented in Table 3.

Table 4 lists the simulation times and resolutions for each of the examples. An example of the relative performance of the different components of the solver is shown in Figure 16. Using 2-5 substeps per frame and 1-3 Newton iterations per substep has been enough to achieve stable results.

9 CONCLUSION

We have presented a novel method for simulating underwater bubbles and realistic wet foam. Our approach improves upon existing methods by providing whitewater that is more naturally coherent with the bulk fluid, while avoiding excessive expert parameter tuning. We achieve these improvements with a holistic principled approach that tracks bubbles from emission, to a guided two-way underwater simulation, and finally to foam as they reach the surface. Our method can efficiently scale to production-level scenarios while providing convincing whitewater effects that hold up even for close-ups.

Limitations. While our approach has generated a number of compelling examples, we have identified several exciting areas for future work. The current approximation only supports wet foams, which covers a large set of applications, but certainly not all of them. Handling dry, stackable foams would be an interesting future extension. Also, our explicit SPH solver experiences instabilities when foam particles have drastically different sizes ($> 5x$). Switching to a more robust, implicit implementation would help eliminate the issue.

The guided bubbles technique works great in open ocean scenarios, but may experience unnatural compression/expansion effects if run in tight spaces over-constrained with collision objects. Essentially, there needs to be enough room for free flow through Dirichlet pressure boundaries for the technique to function properly.

We acknowledge the large scope and implementation complexity of our system, but we believe the benefits far outweigh the costs. We stress that although our method is a system of three primary components (emission, guided bubbles, and wet foam with manifold advection), each individual component would improve existing passively advected white-water methods, and we expect that in many cases interested readers would selectively use our techniques based on their requirements.

We have rendered both bubbles and foam as simple spheres, but we realize this is not practical for production use. We would like to experiment more with volumetric rendering approaches, as well as using texture maps to improve the look of the foam.

Lastly, this work has only considered foam and bubble components of whitewater, but ignored spray and mist, which become increasingly important for faithful simulation of large-scale splashes. We note, however, that our bubble and foam technique can be combined with any of the existing spray and mist techniques, such as the ones presented in [Lesser et al. 2022; Losasso et al. 2008].

ACKNOWLEDGMENTS

We would like to thank Cesar Quijada for his help with the renders, Jon Hertzig for proofreading the later versions of the paper, and Adrien Rollet and Kevin Blom for their unwavering support of this work. We also thank the Simulation and FX departments, and the leadership of Weta Digital for their support in pursuing this topic.

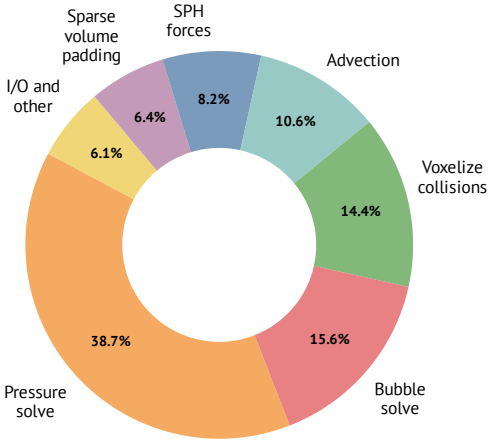

Fig. 16. Relative performance of different components of the solver for the whitewater simulation shown in Figure 2. ©Wētā FX.

Table 4. Simulation times, resolutions, and settings for some of the examples presented in this paper. Numbers of particles, voxels, and time per frame are reported for representative frames that correspond to the ones shown in the figures.

Example	Foam count	Bubble count	Fluid voxels	Steps/frame	Newton iterations	Time/frame	Total frames	Total time
Rocky beach	4.0M	5.5M	18.4M	4	1	1m 28s	476	9h 42m
Submarine	3.2M	1.5M	22.9M	3	3	1m 26s	376	4h 0m
Propellers	297K	347K	665K	5	1	15s	1000	3h 25m
River	1.2M	1.2M	2.7M	5	1	46s	1500	20h 26m
Coupling	0	2.5M	2.0M	2	3	20s	500	1h 17m

REFERENCES

Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2013. Versatile surface tension and adhesion for SPH fluids. 32, 6 (Nov. 2013), 1–8. https://doi.org/10.1145/2508363.2508395

T. B. Anderson and R. Jackson. 1967. Fluid Mechanical Description of Fluidized Beds. Equations of Motion. *Indust. & Eng. Chem. Fund.* 6, 4 (Nov. 1967), 527–539.

Ryoichi Ando and Christopher Batty. 2020. A Practical Octree Liquid Simulator with Adaptive Surface Resolution. 39, 4, Article 32 (jul 2020), 17 pages. https://doi.org/10.1145/3386569.3392460

C. Batty, F. Bertails, and R. Bridson. 2007. A Fast Variational Framework for Accurate Solid-fluid Coupling. *ACM Trans. Graph.* 26, 3, Article 100 (July 2007).

Markus Becker and Matthias Teschner. 2007. Weakly Compressible SPH for Free Surface Flows. *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* 9 (01 2007), 209–217. https://doi.org/10.1145/1272690.1272719

Jan Bender, Dan Koschier, Tassilo Kugelstadt, and Marcel Weiler. 2019. Turbulent Micropolar SPH Fluids with Foam. *IEEE Transactions on Visualization and Computer Graphics* 25 (2019), 2284–2295.

Robert Bridson. 2015. *Fluid Simulation for Computer Graphics, Second Edition.* Taylor & Francis. https://books.google.com/books?id=7MySoAEACAAJ

Oleksiy Busaryev, Tamal K. Dey, Huamin Wang, and Zhong Ren. 2012. Animating bubble interactions in a liquid foam. 31, 4 (Aug. 2012), 1–8. https://doi.org/10.1145/2185520.2185559

Simon Clavet, Philippe Beaudoin, and Pierre Poulin. 2005. Particle-based viscoelastic fluid simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '05*. ACM Press. https://doi.org/10.1145/1073368.1073400

Paul W. Cleary, Soon Hyoung Pyo, Mahesh Prakash, and Bon Ki Koo. 2007. Bubbling and Frothing Liquids. *ACM Trans. Graph.* 26, 3 (jul 2007), 97–es. https://doi.org/10.1145/1276377.1276499

G. Daviet and F. Bertails-Descoubes. 2017. Simulation of Drucker–Prager granular flows inside Newtonian fluids. (Feb. 2017). Working paper or preprint.

Walter Dehnen and Hossam Aly. 2012. Improving convergence in smoothed particle hydrodynamics simulations without pairing instability. *Monthly Notices of the Royal Astronomical Society* 425, 2 (Aug. 2012), 1068–1082. https://doi.org/10.1111/j.1365-2966.2012.21439.x

Luc Deike, W. Kendall Melville, and Stéphane Popinet. 2016. Air entrainment and bubble statistics in breaking waves. 801 (July 2016), 91–129. https://doi.org/10.1017/jfm.2016.372

F F Dunne, F Bolton, D Weaire, and S Hutzler. 2017. Statistics and topological changes in 2D foam from the dry to the wet limit. *Philos. Mag.* 97, 21 (July 2017), 1768–1781.

Douglas Enright, Stephen Marschner, and Ronald Fedkiw. 2002. Animation and Rendering of Complex Water Surfaces. *ACM Trans. Graph.* 21, 3 (July 2002), 736–744. https://doi.org/10.1145/566654.566645

Yun (Raymond) Fei, Christopher Batty, Eitan Grinspun, and Changxi Zheng. 2018. A Multi-scale Model for Simulating Liquid-fabric Interactions. *ACM Trans. Graph.* 37, 4, Article 51 (Aug. 2018), 16 pages. https://doi.org/10.1145/3197517.3201392

Chuyuan Fu, Qi Guo, Theodore Gast, Chenfanfu Jiang, and Joseph Teran. 2017. A Polynomial Particle-in-Cell Method. *ACM Trans. Graph.* 36, 6, Article 222 (nov 2017), 12 pages. https://doi.org/10.1145/3130800.3130878

Ming Gao, Andre Pradhana, Xuchen Han, Qi Guo, Grant Kot, Eftychios Sifakis, and Chenfanfu Jiang. 2018. Animating Fluid Sediment Mixture in Particle-Laden Flows. *ACM Trans. Graph.* 37, 4, Article 149 (jul 2018), 11 pages. https://doi.org/10.1145/3197517.3201309

Carlo Gualtieri, Dragutin Mihailovic, Hubert Chanson, Benoit Cushman-Roisin, Guelfo Doria, Paola Gualtieri, George Kallos, Joe Ackerman, and Borivoi Rajkovic. 2008. *Fluid Mechanics of Environmental Interfaces.*

Francis H. Harlow and J. Eddie Welch. 1965. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *Physics of Fluids* 8 (1965), 2182–2189.

Xiaowei He, H Wang, Fengjun Zhang, Guoping Wang, and Kun Zhou. 2014. Robust Simulation of Small-Scale Thin Features in SPH-based Free Surface Flows. *Life.Kunzhou.Net* 1, 212 (2014), 1–8.

Jeong-Mo Hong, Ho-Young Lee, Jong-Chul Yoon, and Chang-Hun Kim. 2008. Bubbles Alive. In *ACM SIGGRAPH 2008 Papers* (Los Angeles, California) *(SIGGRAPH '08)*. Association for Computing Machinery, New York, NY, USA, Article 48, 4 pages. https://doi.org/10.1145/1399504.1360647

Christopher J Horvath. 2015. Empirical directional wave spectra for computer graphics *(DigiPro '15)*. Association for Computing Machinery, 29–39.

Markus Ihmsen, Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2012. Unified spray, foam and air bubbles for particle-based fluids. 28, 6-8 (April 2012), 669–677. https://doi.org/10.1007/s00371-012-0697-9

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The Affine Particle-in-Cell Method. *ACM Trans. Graph.* 34, 4, Article 51 (jul 2015), 10 pages. https://doi.org/10.1145/2766996

Doyub Kim, Oh-young Song, and Hyeong-Seok Ko. 2010. A Practical Simulation of Dispersed Bubble Flow. *ACM Trans. Graph.* 29, 4, Article 70 (jul 2010), 5 pages. https://doi.org/10.1145/1778765.1778807

A B J Kroezen, J Groot Wassink, and C A C Schipper. 1988. The flow properties of foam. 104, 10 (Oct. 1988), 393–400. https://doi.org/10.1111/j.1478-4408.1988.tb01138.x

Steve Lesser, Alexey Stomakhin, Gilles Daviet, Joel Wretborn, John Edholm, Noh hoon Lee, Eston Schweickart, Xiao Zhai, Sean Flynn, and Andrew Moffat. 2022. Loki: A Unified Multiphysics Simulation Framework for Production. *ACM Trans. Graph.* 41, 4, Article 50 (jul 2022). https://doi.org/10.1145/3528223.3530058

Frank Losasso, Frédéric Gibou, and Ron Fedkiw. 2004. Simulating Water and Smoke with an Octree Data Structure. In *ACM SIGGRAPH 2004 Papers* (Los Angeles, California) *(SIGGRAPH '04)*. ACM, New York, NY, USA, 457–462. https://doi.org/10.1145/1186562.1015745

Frank Losasso, Jerry Talton, Nipun Kwatra, and Ronald Fedkiw. 2008. Two-Way Coupled SPH and Particle Level Set Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics* 14, 4 (2008), 797–804. https://doi.org/10.1109/TVCG.2008.37

J. J. Monaghan. 1992. *Smoothed Particle Hydrodynamics.* Vol. 30. 543–574 pages. https://doi.org/10.1146/annurev.aa.30.090192.002551

D. Morgenroth, S. Reinhardt, D. Weiskopf, and B. Eberhardt. 2020. Efficient 2D Simulation on Moving 3D Surfaces. 39, 8 (Nov. 2020), 27–38. https://doi.org/10.1111/cgf.14098

Matthias Müller, David Charypar, and Markus Gross. 2003. Particle-Based Fluid Simulation for Interactive Applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California) *(SCA '03)*. Eurographics Association, Goslar, DEU, 154–159.

Saket Patkar, Mridul Aanjaneya, Dmitriy Karpman, and Ronald Fedkiw. 2013. A Hybrid Lagrangian-Eulerian Formulation for Bubble Generation and Dynamics. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Anaheim, California) *(SCA '13)*. Association for Computing Machinery, New York, NY, USA, 105–114. https://doi.org/10.1145/2485895.2485912

Andreas Peer, Markus Ihmsen, Jens Cornelis, and Matthias Teschner. 2015. An implicit viscosity formulation for SPH fluids. *ACM Transactions on Graphics* 34, 4 (July 2015), 1–10. https://doi.org/10.1145/2766925

B. Persson and M. Dahlberg. 1994. A Simple Model For Predicting Foam Spread Over Liquids. 4 (1994), 265–276. https://doi.org/10.3801/iafss.fss.4-265

Jos Stam. 1999. Stable Fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 121–128. https://doi.org/10.1145/311535.311548

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. 2013. A Material Point Method for Snow Simulation. *ACM Trans. Graph.* 32, 4, Article 102 (jul 2013), 10 pages. https://doi.org/10.1145/2461912.2461948

Alexey Stomakhin, Joel Wretborn, Kevin Blom, and Gilles Daviet. 2020. Underwater bubbles and coupling. ACM. https://doi.org/10.1145/3388767.3407390

Andre Pradhana Tampubolon, Theodore Gast, Gergely Klár, Chuyuan Fu, Joseph Teran, Chenfanfu Jiang, and Ken Museth. 2017. Multi-Species Simulation of Porous Sand and Water Mixtures. *ACM Trans. Graph.* 36, 4, Article 105 (jul 2017), 11 pages. https://doi.org/10.1145/3072959.3073651

Dominic Vella and L. Mahadevan. 2005. The "Cheerios effect". 73, 9 (Sept. 2005), 817–825. https://doi.org/10.1119/1.1898523

Denis Weaire and Stefan Hutzler. 2001. *The Physics of Foams.* Oxford University Press.

Yonghao Yue, Breannan Smith, Christopher Batty, Changxi Zheng, and Eitan Grinspun. 2015. Continuum Foam. 34, 5 (Nov. 2015), 1–20. https://doi.org/10.1145/2751541

Yongning Zhu and Robert Bridson. 2005. Animating Sand As a Fluid. *ACM Trans. Graph.* 24, 3 (July 2005), 965–972. https://doi.org/10.1145/1073204.1073298