Journal of Computational Physics 228 (2009) 1312–1342

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

HyPAM: A hybrid continuum-particle model for incompressible free-surface flows

Qinghai Zhang [a,c,*], Philip L.-F. Liu [a,b]

[a] School of Civil and Environmental Engineering, Cornell University, Ithaca, NY 14853, USA
[b] Institute of Hydrological and Oceanic Sciences, National Central University, Jhongli, Taiwan
[c] Applied Numerical Algorithms Group, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

ARTICLE INFO

Article history:
Received 8 April 2008
Received in revised form 15 September 2008
Accepted 16 October 2008
Available online 6 November 2008

Keywords:
Free-surface flow
Interface tracking
The Polygonal Area Mapping method
2-Connectedness
Material topology graph
Single-phase decomposition
Passive-response assumption
Hybrid continuum-particle method
Pressure-incremental projection method
Droplet impact
Dam-break problem
Solitary wave propagation

ABSTRACT

Three major issues associated with numerical simulations of complex free-surface flows, viz. interface tracking, fragmentation and large physical jumps, are addressed by a new hybrid continuum-particle model (HyPAM). The new model consists of three parts: (1) the Polygonal Area Mapping method [Q. Zhang, P.L.-F. Liu, A new interface tracking method: the polygonal area mapping method, J. Comput. Phys. 227(8) (2008) 4063-4088]; (2) a new algorithm that decomposes the interested (water) phase into a continuum zone, a buffer zone and a particle zone, based on material topology and graph theory; (3) a 'passive-response' assumption, in which the air phase is assumed to respond passively to the continuum part of the water phase. The incompressible inviscid Euler equations and the equations describing the free fall of rigid bodies are used as the governing equations for the continuum-buffer zone and the particle zone, respectively, and separately. A number of examples, including water droplet impact, solitary wave propagation, and dam-break problems, are simulated for the illustration and validation of HyPAM. It is shown that HyPAM is more accurate and versatile than a continuum-based Volume-of-Fluid model. One major contribution of this work is the single-phase decomposition algorithm, useful for many other hybrid formulations. Neglecting surface tension, viscosity and particle interactions, HyPAM is currently limited to mildly-fragmented free-surface flows with high Reynolds and Weber numbers.

1. Introduction

1.1. Physical background

Free-surface flows are of vast environmental and industrial importance, yet the current understanding of free-surface flows is still poor. Many important physical factors, such as turbulence, surface tension, gravity, viscosity, interfacial mass transfer, effects of surfactants and so on, can come into play and react with each other to produce very complicated flows. At different length-scales and velocity-scales, the behavior of the free surface differs largely. Consequently, there is an enormous range of flow regimes associated with free-surface flows.

Turbulence may be generated at the free surface as in the case of a wave breaker, or, advected from inside the water phase as in the case of a submerged jet. There exists other sources of the free surface turbulence, e.g. the precipitation on the ocean surface [83]. At the presence of strong turbulence, the free-surface features large distortion, which produces droplets,

[*] Corresponding author. Address: School of Civil and Environmental Engineering, Cornell University, Ithaca, NY 14853, USA. Tel.: +1 (607) 3190211.
E-mail addresses: QHZhang@lbl.gov (Q. Zhang), pll3@cornell.edu (P.L.-F. Liu).

0021-9991/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2008.10.029

splashing and/or aeration. Another effect of free surface turbulence is to enhance the mass, momentum and heat transfer between the water and the air.

As opposed to the disturbing nature of turbulence, two main factors that stabilize the free surface are gravity and surface tension; the relative importance of them can be characterized by a length-scale L_*, usually determined from the geometry of the most frequent feature of the free-surface flow. When L_* is large, gravity dominates and keeps the free surface more or less flat. This is the most common state of air–water surface since it applies to a very large percentage of water bodies in the planet such as lakes, rivers and oceans. Sometimes when L_* is small, gravity fails to keep the free surface flat, but surface tension is able to maintain the cohesion of the water surface. A typical example is a weak hydraulic jump (see Fig. 5 of [8]) where the turbulence is not strong enough to generate breakers but sufficient to disturb the surface and create many small smooth knobs on the surface. This regime is called 'knobbly flow' by Brocchini and Peregrine [8]. More often than not, the effect of gravity and surface tension are coupled. As far as vorticity at the free surface is concerned, surface tension generates much larger vorticity than gravity waves do because of the proportionality of the generated vorticity to the local curvature. Thus surface tension contributes to the parasitic capillary rollers and waves often found at the crest of gravity waves. Longuet–Higgins [53] argued that the capillaries and rollers support each other to form a cooperative 'capillary roller' system, with the larger-scale energy supplied by the gravity wave.

The state of the water phase depends on six physical variables $(\rho, v, g, \sigma, L_*, u_*)$, where ρ and v are the density and kinematic viscosity of water, g the gravity, σ the surface tension coefficient, u_* a velocity-scale characterizing *turbulence disturbance*. According to the Buckingham Π theorem, because length, mass and time appear as the three fundamental units, three dimensionless numbers are sufficient and necessary to prescribe the physical system as

$$Re = \frac{u_* L_*}{v}, \tag{1}$$

$$Fr = \frac{u_*^2}{g L_*}, \tag{2}$$

$$We = \frac{\rho u_*^2 L_*}{\sigma}, \tag{3}$$

where the Reynolds number, the Froude number and the Weber number measure the relative importance of turbulent inertial to viscosity, gravity and surface tension, respectively. Following Brocchini and Peregrine [8], free-surface flows with high *Re* can be further classified into four regimes:

(i) The disturbing effect of turbulence is dominant ($Fr \gg 1$, $We \gg 1$). This regime has a very large turbulent velocity-scale.
(ii) The stabilizing effect of gravity is dominant ($Fr \ll 1$, $We \gg 1$). This regime has a very large length-scale.
(iii) The stabilizing effect of surface tension is dominant ($Fr \gg 1$, $We \ll 1$). This regime has very small length-scale and turbulent velocity-scale.
(iv) Both gravity and surface tension are more important than the turbulence ($Fr \ll 1$, $We \ll 1$). This regime has a very small velocity-scale.

In the case of $Fr \ll 1$ and $Re \ll 1$, two viscous boundary layers were identified by Hunt and Graham [34] near the free surface: an inner viscous layer and an outer source layer, also called the 'blockage' layer. For a co-moving rigid boundary, their analysis shows that the thickness of the inner viscous layer scales with the square root of the local Reynolds number. In the thicker blockage layer, turbulence intensity is redistributed such that the normal component decreases and the lateral and stream-wise components are amplified. Later Direct Numerical Simulation (DNS) by Shen et al. [77] confirmed the existence of these two layers and further examined the their effects on the turbulence statistics of length-scales, Reynolds-stress balance and enstrophy dynamics. Compared to a free-slip flat plate, the free surface was found to considerably reduce the pressure–strain correlation [77] and eddy viscosity [75]. However, if $Re \gg 1$, the length-scale of the inner viscous layer is much smaller than the length-scale of the interface distortions and the blockage layer can be much reduced or even neglected when the stabilizing effects of gravity and surface tension are insufficient to constrain the vertical fluctuations [8]. Thus viscous effects are relatively unimportant if $Re \gg 1$.

When free surface turbulence is just strong enough to break up the free surface, the movement of a water droplet can be well described by an inviscid trajectory; these flow regimes are identified as 'ballistic flow' and 'splashing flow' in [8]. However, if the air motion is significant, the interaction between the escaping water drops and the surrounding air becomes important. For example, big water droplets can break up to smaller ones, the drag and lift forces from the air can influence the water droplets substantially and even create a flow inside a water drop [81]. Another related phenomenon is air entrainment, with the two basic mechanism as local aeration and interfacial aeration. Local aeration is usually caused by an impinging jet on a receiving pool through by a Couette flow mechanism [11] and the details of the aeration vary with the jet velocities. Interfacial aeration happens when the disturbing effect of turbulence overcomes gravity and surface tension. It is often coupled to a change of bottom geometry, such as spillway flows and 'white water' down a mountain stream. Splashing and air bubble entrainment contribute significantly to water–air mass transfer since the total surface area of many tiny bubbles or droplets is much greater than that of a flat free surface. Indeed, wave breaking is one of the most important mass transfer mechanisms in the ocean [83,56]. In fact, mass and heat transfer happens even at a smooth free surface, through the viscous boundary layer [51,52] or evaporative convection [24].

Apart from the above physical processes, free-surface flows are also influenced by their chemical constitutions. The effects of surfactants has long been a research field [68,76]. Scott [72] demonstrated that just 1% of salt in water makes a substantial difference to the entrapped bubble size.

1.2. Fundamental issues and main objectives

As discussed in the previous section, there are vast physical aspects of free-surface flows. It is extremely difficult, if not impossible, to develop a single numerical model that incorporates all these aspects appropriately. Instead, we address three major issues in this paper as a starting point towards a more general framework for free-surface flows.

I. *Interface tracking.* The location of interface is a part of the solution and affects the simulation of the velocity field.
II. *Large jumps in physical variables.* Physical quantities such as density and viscosity have very large gradients across the interface, often represented by step functions. In addition, the density ratio of the two fluids is large.
III. *Fragmentation*. For complex free-surface flows, the interface is no longer a smoothly connected material surface. This process often features a topology change of a compact macroscopic fluid volume and its subsequent dispersion into stable droplets of smaller scales [86].

These three issue are chosen for several reasons. The most fundamental question for free-surface flow simulation is: where is the free surface? Compared to this, all the other physical aspects are secondary in importance because they all depend on the interface locus. In a numerical simulation, the interface locus is uniquely determined once the velocity is given and an interface tracking method is chosen. It is well known that VOF methods have large numerical diffusion, particularly when an interface contains high curvatures [96]. Furthermore, the so-called ‘momentum-capturing’ method [12] used in many VOF methods approximates the velocity field near the interface by a zeroth order extrapolation from inside the water phase, as discussed in more details in Section 3.2. Consequently, the interface locus results of VOF methods contain large errors; so does numerical modeling of other physical aspects. Alternatives are thus desirable to address the first two issues more accurately.

Fragmentation is ubiquitous in complex free-surface flows. In addition, they often causes instability for numerical simulations. In numerically diffusive methods such as VOF methods, stability is obtained at the price of smearing out fragmentation. Thus VOF-based models are not appropriate for simulating fragmentation and alternatives have to be sought. The numerical aspects of these issues will be discussed in more details in the next section.

The main objectives of this work are to

(1) develop a numerical model that produces accurate results of both the interface location and the underlying flow field,
(2) propose a single-phase decomposition algorithm that separates the continuum zone from the fragmentation zone so that continuum-based methods and particle-based methods can be applied separately.

To focus on the main objectives, we first neglect surface tension and viscous effects so as to confine ourselves to flow regimes (i) and (ii), i.e., free-surface flows with $Re \gg 1$ and $We \gg 1$. To measure the relative importance of gravity to surface tension, the Bond number is defined as

$$Bo = \frac{\rho g L_*^2}{\sigma}. \tag{4}$$

$Bo = 1$ yields a critical length-scale $L_*^\sigma \approx 2.7$ mm, below which surface tension dominates. It is important to point out that rather than having a single length-scale, realistic free-surface flows often feature a cascade of length-scales, each of which characterizes a dominant physical factor. Apparently, L_*^σ is the smallest physical length-scale at the free surface in the physical system prescribed by Re, Fr and We. In our numerical model, we assume the grid size h is bigger than L_*^σ, as a justification of neglecting surface tension. Inevitably, this introduces an element of ‘modeling’; the volume adjusting algorithms in interface tracking is such an example, as discussed in Section 2.2.

Because this paper concerns only two-dimensional space, many interesting physics existing only in three-dimensional space are excluded, such as vortex connection [95] and vortex pair modulation [89] at the free surface.

1.3. Computational background

Currently, numerical models for free-surface flows fall into two broad categories: grid-based continuum methods and mesh-free particle methods.

In the grid-based continuum methods, the interface tracking is often done by volume-of-fluid (VOF) methods [69] or level-set methods [74]. Despite their previous successes, these two methods are numerically diffusive and incapable of tracking interface with singular points and high curvatures (i.e. sharp corners). Recently, an alternative, the Polygonal Area Mapping (PAM) method [96], representing material areas as piecewise polygons and tracking the interface via discrete polygon-clipping algorithms, has been developed. In addition to its independence from mesh topology, the PAM method has very little numerical diffusion and is highly accurate even in tracking interface singularities.

Various formulations to address the issue concerning large physical jumps have been proposed. Based on the work of Kothe [39] and Lemos [41], Lin and Liu [45] presented a wave-breaking model called COBRAS, in which a two-step projection method [13,14] is coupled to a VOF method [32]. The large physical jumps are treated implicitly by performing computations across the interface and setting the velocity and pressure of air to zero. An improved version of COBRAS was used to study nonlinear liquid sloshing with broken free surfaces [46].

Instead of treating the air phase as a void, one can restrict the computational domain to the water phase and apply the boundary conditions *explicitly* on the moving free surface. This removes the large physical jumps and saves computational resources since the air phase does not have to be involved in the calculation. With a level-set method as the interface tracker, Watanabe [87] implemented such a formulation using the Constrained Interpolation Profile (CIP) method [93] for the treatment of the advection terms and an irregular-star method [10] for the pressure Poisson solver. Since the mesh is fixed, the free surface boundary conditions must be satisfied by extrapolation from inside the water phase.

In contrast to fixed-mesh methods, mesh-moving methods [26,88,33,50] deform the mesh to follow the interface motion. This yields an easy solution of the large physical jumps at the expense of introducing additional re-meshing procedures; see [18] for an adaptive tetrahedral mesh, [65] for an unstructured staggered mesh and [94] for a boundary-fitted mesh by orthogonal mapping.

If the free surface topology is simple, e.g., it can be described by a single-valued function, the dimension of the problem can be reduced by using either the shallow water equations [19] or boundary integral methods [98,16]. For steady state flows, one can solve the problem by the domain deformation method [26,91] or even simplify the problem to Laplace equations [55,64] by neglecting viscous effects.

For all of the above formulations, the fragmentation issue is either avoided or smeared out by numerical artifacts, e.g., boundary integral methods become unstable and VOF methods are not suitable for modeling fragmentation, which will be discussed in more detail in Section 5.4.1. Indeed, if the fragments are smaller than a single computational cell, a numerical model based on the continuum hypothesis is incapable of describing their dynamics, because of the limited resolution. Under this circumstance, mesh-free particle methods are more appropriate, since they are fundamentally linked to the particle physics, particularly in the micro- and nano-scale.

Various particle methods have already been applied to free-surface flows, including the smoothed particle hydrodynamics (SPH) [58,59], the moving-particle semi-implicit (MPS) method [38,37], the Lattice Boltzmann (LB) method [35,27], and the natural element method [28]. Although more appropriate in addressing the fragmentation issue and more convenient in avoiding difficulties in tracking a free surface with large jumps in physical variables, they encounter many unexpected difficulties. While some of these difficulties such as the stability issue have already been improved by new particle evolution equations [15], new renormalization-based formulations [61] or new kernel functions [3], other difficulties, such as accurately setting boundary conditions, remain challenging. Also, their adaptivity to multi-scale problems is often undermined by severe particle distortion that may introduce spurious scales [40].

Clearly, hybrid models, aiming to combine the best features of the two methods while eliminating their disadvantages, are needed. Liu et al. [48] coupled the MPS method to a conservative finite volume method, in which one phase is represented by the particles and the other phase by the rest of the domain without particles. Different phases are treated as one fluid with variable material properties. A Poisson pressure equation is first solved to obtain a divergence free velocity field. Then, in order to target the desired particle number density, a second Poisson equation is solved again to artificially change the particle positions. As noted in [54], this might yield non-physical solutions for a stationary flow field with variable densities. For computer graphics applications, Losasso et al. [54] coupled SPH to a two-step projection method on a fixed mesh, with a hybrid particle-level set method [22,23] for interface tracking. They solved the Poisson pressure equation only once to compute the divergence-free velocity field and introduced the notion of 'particle slip' to extend their SPH solver to simulate phenomena such as mixtures of spray and air. A number of interesting examples including the crashing of ocean waves against a lighthouse and beach are simulated, but no rigorous physical validation was conducted. Several other hybrid methods for physically based rendering are reported in [25,73]. A more physically oriented hybrid particle-continuum method is presented by Williams et al. [90] for fluctuating hydrodynamics. The particle algorithm, direct simulation Monte Carlo (DSMC), is based on the Boltzmann equation while the continuum algorithm is based on the Landau–Lifshitz Navier–Stokes (LLNS) equations, which incorporate thermal fluctuations into macroscopic hydrodynamics by stochastic fluxes.

In the aforementioned hybrid methods, the water phase is represented *both* by the particles and the background mesh. An alternative formulation, currently restricted to micro- and nano-scale flows [60,20], decomposes a *single* phase into a continuum zone and a particle zone, described by Navier–Stokes (NS) equations and molecular dynamics (MD), respectively. The consistency of the two zones is enforced in an overlapping region by various techniques such as Schwarz iterations [40]. The single-phase decomposition has limited the application to flows with static interfaces [17,47] or steady states [71]. Indeed, if the interface between these two zones moves arbitrarily, it is difficult to achieve the decomposition in a physically consistent manner. This has impeded the generalization of this formulation to macro-scale multi-phase flows. One of the main purposes of this paper is to resolve this difficulty. Even for a DNS, a decomposition algorithm is necessary because the length-scale of a local topology change is much smaller than the Kolmogorov length-scale since the process of pinching involves breaking up a zero-sized connecting ligament [29]. At this resolution, the continuum assumption of Navier–Stokes equations may also break down. Thus a single-phase decomposition algorithm is highly desirable.

In our new model, HyPAM, the topology of a single phase is modeled by a material graph. Then the notion of '2-connectedness' is employed in Section 2 to decompose the phase into a continuum zone, a buffer zone and a particle zone, in a way consistent with the underlying physics and the finite difference scheme. The issue concerning the large jump of physical variables across the interface is addressed by the *passive-response* assumption in Section 3. Numerical details to enforce this assumption are given in Section 4. A number of tests are performed in Section 5 before we draw conclusions in Section 6.

2. Material topology

Only for the purpose of exposition, we assume structured rectangular grids with uniform spacing $\Delta x = \Delta y = h$ and uniform time stepping $t_n = t_0 + n\Delta t$ unless explicitly specified otherwise. Both the PAM method [96] and the single-phase decomposition algorithm to be described herein can be applied to unstructured grids without any modifications.

2.1. A Voronoi-diagram viewpoint

Many computational grids with convex cells, either structured or unstructured, can be viewed as a *Voronoi diagram* [62]. Let $\Omega \subset \mathbb{R}^d$ be the computational domain, $\mathbb{C} = \{c_{i,j} | c_{i,j} \in \Omega\}$ a finite set of points. These points are also called the *sites*; assign every point in Ω to its nearest site such that all the points assigned to $c_{i,j}$ are at least as close to $c_{i,j}$ as to any other site:

$$\mathcal{C}_{i,j} = \{\mathbf{x} : |c_{i,j} - \mathbf{x}| \leqslant |c_{\ell,m} - \mathbf{x}| \forall c_{\ell,m} \in \mathbb{C}\}, \tag{5}$$

where $\mathcal{C}_{i,j}$ is called a *Voronoi polygon*. Obviously some points do not have a unique nearest site. The Voronoi diagram $\mathbb{V}(\mathbb{C})$ is the set of all points that have more than one nearest sites. $\mathbb{V}(\mathbb{C})$ partitions the domain into Voronoi polygons.

For a structured rectangular mesh, the sites are precisely the cell centers, the Voronoi polygons the cell polygons, and the Voronoi diagram the mesh itself. Because of (5), the sites are often chosen as the locations where discrete values of a physical quantity are stored. A specific form of its distribution is then assumed inside a Voronoi polygon, such as a constant distribution, a linear distribution and so on. In other words, the sites, serving as representatives of all the points in the domain, together with the assumed distribution inside each Voronoi polygon, form the geometric base of discrete approximations to continuous distributions of physical quantities. This is one of the main topological features of a continuum method.

We define the *neighborhood* of a cell $[i,j]$ to be the union of the Voronoi polygons that have non-empty intersections to $\mathcal{C}_{i,j}$:

$$\mathcal{N}_{i,j} = \bigcup_{\ell,m} \mathcal{C}_{\ell,m}, \qquad \forall \mathcal{C}_{\ell,m} \cap \mathcal{C}_{i,j} \neq \emptyset. \tag{6}$$

Cell $[i,j]$ is called an *interface cell* if $\mathcal{C}_{i,j}$ contains more than one material, otherwise it is called *a pure cell*. An *interface candidate cell* is a cell whose neighborhood contains an interface cell.

2.2. Polygonal Area Mapping

The details of the PAM method [96] can be found in [96]; it is briefly summarized here only for completeness and the ease of discussions.

In the PAM method, material areas are represented by a set of *simple* polygons (SSP),

$$\mathcal{M} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{s_p}, \mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_{s_h}\}, \tag{7}$$

where $\mathcal{P}_i$ represents a 'positive' area, $\mathcal{H}_j$ a 'negative' area (a hole) and s_p, s_h are the numbers of them, respectively. A simple polygon is assumed to be 'well-shaped' so that the interior of the triangle formed by any three consecutive vertices are either a subset of the polygon or disjoint to the polygon:

$$p_i \text{ is a convex vertex of } \mathcal{P} \Rightarrow q \in \mathcal{P} \quad \forall q \in \Delta(p_i, p_{i-1}, p_{i+1}), \tag{8a}$$

$$p_i \text{ is a concave vertex of } \mathcal{P} \Rightarrow q \notin \text{interior}(\mathcal{P}) \quad \forall q \in \Delta(p_i, p_{i-1}, p_{i+1}), \tag{8b}$$

where $\varDelta(p_i, p_{i-1}, p_{i+1})$ denotes the triangle formed by vertices p_i, p_{i-1}, p_{i+1}. In addition, the following requirements are also enforced:

$$i \neq j \Rightarrow \mathcal{P}_i \cap \mathcal{P}_j = \emptyset, \tag{9a}$$

$$i \neq j \Rightarrow \mathcal{H}_i \cap \mathcal{H}_j = \emptyset, \tag{9b}$$

$$\mathcal{H}_i \in \mathcal{M} \Rightarrow \exists \mathcal{P}_j \in \mathcal{M}, \ s.t. \mathcal{H}_i \subset \mathcal{P}_j, \mathcal{P}_j \setminus \mathcal{H}_i \text{ is not a simple polygon}, \tag{9c}$$

$$\forall \mathcal{P}_i, \mathcal{H}_j \in \mathcal{M}, \ \text{no three consecutive vertices are collinear}, \tag{9d}$$

so that the area can be *uniquely* represented by

$$\mathcal{M} = (\cup_{i=1}^{s_p} \mathcal{P}_i) \setminus (\cup_{j=1}^{s_h} \mathcal{H}_j), \tag{10}$$

and the total volume of $\mathcal{M}$ can be easily calculated by

$$\|\mathcal{M}\| = \sum_{i=1}^{s_p} \|\mathcal{P}_i\| - \sum_{j=1}^{s_h} \|\mathcal{H}_j\|, \tag{11}$$

where $\|\cdot\|$ denotes the volume of a compact area.

Using the concept of SSP, a sharp interface tracking problem is formulated as the determination of $\mathcal{M}(T)$, the area occupied by the water phase, from $\mathcal{M}(t_0)$, the initial area of the water phase, and a given velocity field $\mathbf{u}(\mathbf{x}, t)$ for $t \in [t_0, T]$.

A particle $p(t_n)$ is mapped to its image $\overrightarrow{p(t_n, \delta t)} = p(t_n + \delta t)$ and preimage $\overleftarrow{p(t_n, \delta t)} = p(t_n - \delta t)$ by following its pathline through $\mathbf{u}(p)$ and $\mathbf{u}_B(p) = -\mathbf{u}(p)$:

$$\overrightarrow{p(t_n, \delta t)} = p(t_n + \delta t) = p(t_n) + \int_{t_n}^{t_n+\delta t} \mathbf{u}(p(t), t)\mathrm{d}t \tag{12}$$

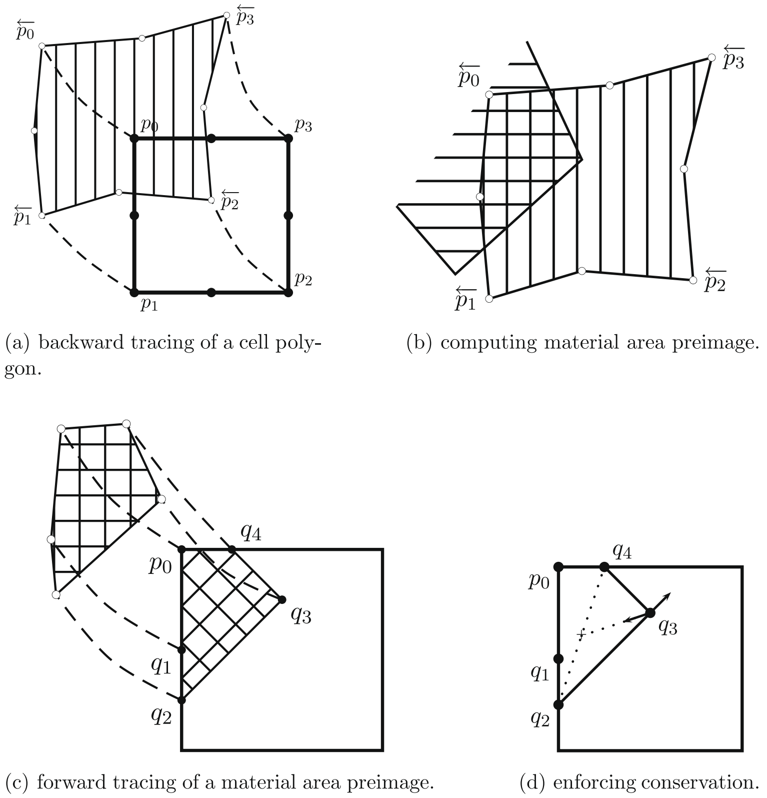
$$\overleftarrow{p(t_n, \delta t)} = p(t_n - \delta t) = p(t_n) - \int_{t_n}^{t_n-\delta t} \mathbf{u}_B(p(t), t)\mathrm{d}t, \tag{13}$$

A SSP has similar notations, e.g., the preimage of the cell polygon at t_{n+1} is denoted by $\overleftarrow{\mathcal{C}_{i,j}^{n+1}}$, a shorthand notation for $\overleftarrow{\mathcal{C}_{i,j}(t_{n+1}, \Delta t)}$.

At time t_n, the PAM method advances the interface by applying the following steps to each interface candidate cell, whose definition entails that the Courant number be smaller than one.

(a) Solve $\frac{Dx}{Dt} = \mathbf{u}_B$, with $\mathcal{C}_{i,j}$ as the initial condition, for the cell preimage $\overleftarrow{\mathcal{C}_{i,j}^{n+1}}$, shown in Fig. 1(a) and (b) as the vertical hatches;

(b) Calculate the material preimage $\overleftarrow{\mathcal{M}_{i,j}^{n+1}}$ using a polygon-clipping algorithm [85] by intersecting $\overleftarrow{\mathcal{C}_{i,j}^{n+1}}$ with $\mathcal{N}_{i,j,M}^{n}$, the material area in $\mathcal{N}_{i,j}$ at t_n, shown as the horizontal hatches in Fig. 1(b); $\overleftarrow{\mathcal{C}_{i,j}^{n+1}}$ contains all the particles to be advected



(a) backward tracing of a cell polygon.

(b) computing material area preimage.

(c) forward tracing of a material area preimage.

(d) enforcing conservation.

Fig. 1. The polygonal area mapping (PAM) method. (a),(b),(c), and (d) correspond to the four sub-steps of the PAM method, respectively. Solid dots (•) and hollow dots (○) represent image points and preimage points, respectively. Dashed lines denote pathlines.

into cell $[i,j]$ at the end of the time step, these particles may or may not be of the material type being tracked. $\mathcal{N}^n_{i,j,M}$ contains all the particles of the tracked material at the beginning of the time step. $\overleftarrow{\mathcal{C}^{n+1}_{i,j}} \cap \mathcal{N}^n_{i,j,M}$ thus yields all the tracked material particles that will end up inside $\mathcal{C}_{i,j}$ at the end of this time step;

(c) Solve $\frac{Dx}{Dt} = \mathbf{u}$, with $\overleftarrow{\mathcal{M}^{n+1}_{i,j}}$ as the initial condition, for $\mathcal{M}^{n+1}_{i,j}$, shown as the 45° cross hatches in Fig. 1(c);

(d) Change $\mathcal{M}^{n+1}_{i,j}$ within $\mathcal{C}_{i,j}$ to enforce mass conservation and representation invariants of concrete data structures, an example is shown in Fig. 1(d) and the algorithms for general geometric configuration is detailed in [96].

To ensure stability, we introduce a global constant $n_{\max}$ as the maximal number of vertices allowed in a simple polygon. With the significance of a vertex p_i being measured by $\|\Delta(p_i, p_{i+1}, p_{i-1})\|$, the least significant vertices in a polygon are deleted until the number of vertices does not exceed $n_{\max}$. Should any deletions occur, the volume-adjusting algorithms in (d) are invoked to ensure mass conservation. For all the numerical tests performed in Section 5, $n_{\max}$ is set to 10; this is also the reason why eight points are tracked in Fig. 1(a).

Another significance of the volume-adjusting algorithms in (d) concerns an implicit modeling of surface tension. A material polygon is adjusted in a way such that the total number of vertices is non-increasing and, consequently, the length of the interface decreases. Therefore, although surface tension is neglected in HyPAM, its effects are implicitly incorporated in the numerical model. One might argue that this implicit modeling of surface tension is not accurate; indeed this is true because the limit of the computational resource imposes $h/L^{\sigma}_{*} > 1$. An explicit implementation of surface tension is more appropriate as a future improvement in the case of $h/L^{\sigma}_{*} < 1$.

Since discrete Boolean operations instead of calculus are used in the interface tracking, the PAM method has very little numerical diffusion and is highly accurate. However, incorporating the PAM method into a hybrid continuum-particle model requires a definition of the single-phase connectedness. To this end, we employee the concept of '2-connectedness' in graph theory. The readers are referred to [21] for a mathematical treatise and [31] for a computational resource on graph theory.

2.3. Material topology graph

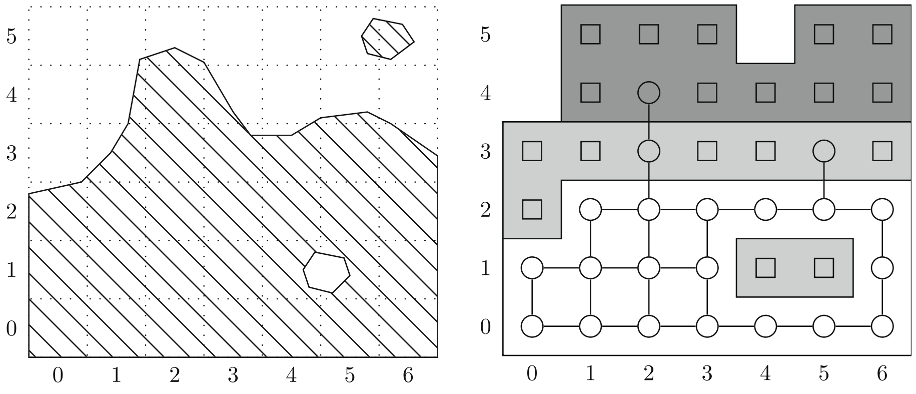
A graph is a mathematical structure with a pair of sets, $G = (V, E)$, where V is the set of *vertices* and E is the set of *edges*. We say G is a graph *on* V. The elements of the edge set are two-element subsets of V, $e = \{v_1, v_2\} \in E$, usually written as (v_1, v_2). v_1 and v_2 are the *source* and the *target* of e, respectively. A vertex v is *incident* with an edge e if $v \in e$; then e is an edge *at* v. G can be either *undirected* or *directed* depending on whether the elements of E are ordered or not. A graph is *simple* if it has neither self-loops (edges joining a vertex to itself) nor multi-edges (edges with identical endpoints). We only use simple undirected graphs in this work.

We define $G \cup G' = (V \cup V', E \cup E')$ and $G \cap G' = (V \cap V', E \cap E')$. G and G' are *disjoint* if $G \cap G' = \emptyset$. G' is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$. $G - V'$ is defined by deleting all vertices in $V' \cap V$ and their incident edges in E.

After the domain is partitioned by the computational grids (see Fig. 2(a) for an example), the topology of a single phase W can be modeled by a simple undirected graph G_W on $\mathbb{C}_W$ where $\mathbb{C}_W \subseteq \mathbb{C}$ and $\mathbb{C}$ is the set of Voronoi sites. Given $\mathcal{M}$, the SSP of W in the domain $\Omega \subset \mathbf{R}^d$, this topology graph can be constructed by the following rules

(i) A vertex $c_{i,j} \in \mathbb{C}_W$ if and only if

$$\mathcal{C}_{i,j} \cap \mathcal{M} = \mathcal{C}_{i,j} \tag{14}$$



(a) Material polygons

(b) Material topology graph

Fig. 2. Representation of material topology by a graph G_W on a rectangular mesh. The set of vertices consists of only pure cells ('○'), excluding interface cells ('□'). Edges exist only between pure cells that are face neighbors. The white area represents the continuum zone, the light-gray area the buffer zone, and the dark-gray the particle zone.

(ii) For the vertices in $\mathbf{C}_W$, an edge $(c_{i,j}, c_{\ell,m}) \in E_W$ if and only if the two cells are face neighbors. Mathematically, this requires that the *affine hull* of $\mathcal{C}_{i,j} \cap \mathcal{C}_{\ell,m}$ be of $d-1$ dimension. On two-dimensional rectangular structured grids, this is equivalent to

$$|i-\ell| + |j-m| = 1 \tag{15}$$

The counterpart of (15) for unstructured grids is often related to the specific data structure in use.

According to (14), a cell is a vertex of G_W if and only if it is a pure cell of material W. From the Voronoi diagram viewpoint, the site of an interface cell is no longer an appropriate representative for all the points of different material types inside the Voronoi polygon. Furthermore, the distribution of a physical quantity in an interface cell deviates largely from that in a pure cell because of the jumps in that quantity and its derivatives. Therefore, an interface cell should not be in the material graph that models the topology of a single phase.

(15) implies that two cells have to share a common face to qualify for an edge between them. This reflects the fact that the finite difference discretization of the governing equations concerns only cells that are local face-neighbors. If any corner neighbors, such as $[i+1, j+1]$ or $[i-1, j-1]$, were involved in derivative evaluations at cell $[i,j]$, there would exist edges connecting cell $[i,j]$ to these corner neighbors. As an example of this graph model, a material SSP is shown in Fig. 2(a) and its corresponding graph is shown in Fig. 2(b).

2.4. The single-phase decomposition algorithm

A *path* is a special kind of graph $P = (V, E)$ with the form

$$V = \{v_0, v_1, \ldots, v_k\}, \quad E = \{(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)\} \tag{16}$$

where v_i's are all distinct and the vertices $v_1, \cdots, v_{k-1}$ are the *inner* vertices of P. Given two vertex sets A and B, P is called an $A-B$ *path* if $V(P) \cap A = \{v_0\}$ and $V(P) \cap B = \{v_k\}$. Two or more paths are *independent* if they do not share any inner vertices.

A graph G is *connected* if there exists a path between any two of its vertices. A maximal connected subgraph of G is called a *component* of G. V' *separates* G if $G - V'$ has more components than G, then V' is called a *vertex-cut* or *separator*. A *cut-vertex* is a vertex-cut consisting of only a single vertex. Similarly, an *edge-cut*, E', is a set of edges such that removing these edges from G results in an increase of the number of components. A *bridge* or a *cut-edge* is an edge-cut consisting of a single edge. In Fig. 2(b), G_W is connected. An vertex-cut of G_W is $\{c_{4,2}, c_{4,0}\}$ and the set of all cut-vertices is $\{c_{2,2}, c_{2,3}, c_{5,2}\}$. An edge-cut of G_W is $\{(c_{4,2}, c_{5,2}), (c_{5,0}, c_{6,0})\}$ and the set of all bridges is $\{(c_{2,2}, c_{2,3}), (c_{2,3}, c_{2,4}), (c_{5,2}, c_{5,3})\}$.

From the construction rules of the material topology graph G_W, the existence of a A—B path implies that physical properties at A and B depend on each other, or, influence each other. If there is only one independent path P between A and B, at least one inner vertex of P has to be a cut-vertex, implying that a material polygon containing A and B has a 'bottleneck' with its width roughly the same as the grid size. In this case, one cannot be sure about the connectedness because of the numerical uncertainty imposed by the grid size. Furthermore, numerical robustness is often undermined by a bottleneck. Therefore, we require there be more than one independent path between A and B, this leads to the concept of k-connectedness.

$G = (V, E)$ is k-*connected* (k is a positive integer) if $|G| > k$ and $G - V'$ is connected for every set $V' \subseteq V$ with $|V'| < k$, where $|G| = |V|$ is the *order* of a graph, i.e., the number of its vertices. Two more intuitive definitions of k-connectedness are

- At least k vertices have to be removed in order to disconnect a k-connected graph.
- A graph is k-connected if any two of its vertices can be joined by k independent paths.

It is well known that the above two definitions are equivalent since they are dual aspects of the same property.

A maximal connected subgraph without a cut-vertex is called a *block*. Every block of a graph is either a maximal 2-connected subgraph, or a bridge, or an isolated vertex. The converse is also true. By their maximality, different blocks of G share at most one vertex, which must be a cut-vertex. It follows that G is the union of its blocks. *Block decomposition* of G is equivalent to finding all cut-vertices of G, which depends heavily on Depth-First-Search (DFS) [1,4,84]. Let tree T be the result of applying DFS to a connect graph G, the root of T is a cut-vertex of G if and only if the root has more than one child in T; a non-root v of T is a cut-vertex of G if and only if v has a child v' such that no descendent of v' is joined to a proper ancestor of v by a non-tree edge [31].

A vertex of G might be in different blocks whereas any edge of G belongs to a unique block. Thus the effect of a block decomposition algorithm is that each edge being assigned a number such that all edges with the same number form a distinct block. A linear block decomposition algorithm proposed by Tarjan [82] and the Boost graph library [79] are used in this work.

We formalize single-phase decomposition in Algorithm 1, which uses the following routines:

- `AddVertex`(v, G) add v as a vertex into G;
- `AddEdge`(v_1, v_2, G) add an edge between v_1 and v_2;
- `BlockDecomposition`(G) finds all the blocks of G, appends a number to each edge of G and returns the number of blocks of G.

Algorithm 1: Single-phase decomposition. $c_{i,j}, \mathcal{C}_{i,j}$ and $\mathcal{M}_{i,j}$ are the label, shell polygon and material polygon of cell $[i,j]$, respectively. $\mathcal{N}_{i,j}$ is the neighborhood of cell $[i,j]$ defined in (6).

```
Data: A SSP $\mathcal{M}$ of material $W$ in the domain; a Voronoi diagram $\mathbb{V}(\mathbb{C})$
      as the partition of the domain;
Result: Each cell that contains the material is colored to 'WHITE' if it
        is 2-connected; 'GRAY' if it is in the neighborhood of a WHITE
        cell; 'BLACK' otherwise.

1  $G = (V, E) \leftarrow$ an empty graph
2  foreach $c_{i,j} \in \mathbb{C}$ and $\mathcal{C}_{i,j} \cap \mathcal{M} \neq \emptyset$ do
3      color[$c_{i,j}$] $\leftarrow$ BLACK
4      if $\mathcal{C}_{i,j} \cap \mathcal{M} == \mathcal{C}_{i,j}$ then
5          AddVertex ($c_{i,j}, G$)
6      end
7  end
8  foreach $c_{i,j} \in V$ do
9      foreach $\mathcal{C}_{\ell,m} \subset \mathcal{N}_{i,j}$ and $c_{\ell,m} \in V$ do
10         if $c_{i,j}$ and $c_{\ell,m}$ are face neighbors then
11             AddEdge ($c_{i,j}, c_{\ell,m}, G$)
12         end
13     end
14 end
15 $n_{\text{block}} \leftarrow$ BlockDecomposition ($G$)
16 $n^{\ell}_{\text{edge}} \leftarrow$ the number of edges in the $\ell$-th block ($1 \leq \ell \leq n_{\text{block}}$)
17 foreach $e \in E$ do
18     $m \leftarrow$ the label of the block that contains $e$
19     if $n^{m}_{edge} > 2$ then
20         color[Source ($e$)] $\leftarrow$ WHITE
21         color[Target ($e$)] $\leftarrow$ WHITE
22     end
23 end
24 foreach $c_{i,j} \in \mathbb{C}$ and color[$c_{i,j}$]==BLACK do
25     foreach $\mathcal{C}_{\ell,m} \subset \mathcal{N}_{i,j}$ and color[$c_{\ell,m}$]==WHITE do
26         if $\mathcal{M}_{i,j} \cap \mathcal{M}_{\ell,m} \neq \emptyset$ then
27             color[$c_{i,j}$] $\leftarrow$ GRAY
28             break
29         end
30     end
31 end
```

Algorithm 1 has four parts. Lines 1–14 assemble the material topology graph G_W based on the rules described in Section 2.3. Lines 15–16 decompose G_W into its blocks. Lines 17–23 color all the 2-connected cells white. Note that line 19 uses the simple fact that a 2-connected component must have least three edges. For a cell not 2-connected, lines 24–31 color it gray if its material polygon is geometrically connected to that of a 2-connected cell. The other cells are colored black.

It is important to point out why a Voronoi diagram, not a general domain partitioning, is used as the input. The main reason is that *the partition of the domain and the single-phase decomposition are coupled*. Given the SSP of a material in the domain, the type of a cell varies from one domain partitioning to another. As argued in Section 2.3, the site of an interface cell is not an appropriate representative for all the points in the cell because they are of different material types and the jumps of the physical quantities invalidate the assumed distribution form. In an extreme case that the domain is partitioned into thin and long stripes, there might be no pure cells and consequently no continuum zone. Therefore, a specific type of domain partitioning conducive to the single-phase decomposition is needed.

Numerical simulations store quantities at some chosen discrete points (sites) with an assumed distribution form. The coverage of a cell is then *implicitly* determined by the sites, at least in finite difference schemes. In other words, given a set of discrete points as locations of the quantities, the method to construct the 'cells' from the sites and the single-phase decomposition algorithm *uniquely* determine the continuum zone and the particle zone. In constructing the 'cells' from the sites, Voronoi diagram, specifically (5), is the best choice in the context of single-phase decomposition. Although using Voronoi diagram in formulating Algorithm 1 incurs a loss of generality, this loss of generality is good because many inappropriate ways of domain partitioning are excluded.

Another benefit of Voronoi diagram is the independence of Algorithm 1 from mesh topology. For unstructured grids, only the subscripts in Algorithm 1 need to be changed.

It is trivial to verify the complexity of Algorithm 1 as $O(\frac{1}{h^2}\log\frac{1}{h^2})$, if an adjacency list and a map are chosen as the concrete data structures. In fact, adjacency list is an excellent choice since there are at most four incident edges at any vertex, due to the local topology feature of the material graph.

Referring to Algorithm 1, we define the *continuum zone* as the union of all the 'white' (2-connected) cell polygons, the *buffer zone* the union of all the 'gray' cell polygons enclosing the continuum zone, and the *particle zone* the union of all the 'black' cell polygons, which contains the rest of material W. They corresponds to white, light-gray and dark-gray areas in the example of Fig. 2(b).

Algorithm 1 is very useful for hybrid methods. For example, one can employ a boundary integral method or an irregular-star method for the continuum zone, apply boundary conditions in the buffer zone and use a particle method to simulate the particle zone. Alternatively, one may make use of the buffer zone to ensure a smooth transition from the continuum zone to the particle zone [20,60]. The 2-connectedness of the material SSP in the continuum zone guarantees well-posedness of an elliptic-type governing equation because any cell in the continuum zone is inside a circle. Furthermore, when extrapolation from one phase into another is desirable, the 2-connectedness of the continuum zone guarantees enough source data.

Algorithm 1 is also useful for mesh-moving methods. After all, one major advantage of the mesh-moving methods is to ensure 2-connectedness of single-phase topologies near the interface. However, these methods still suffer from instability when the interface are fragmented. Thus they can also benefit from Algorithm 1 for better stability.

More generally, *the single-phase decomposition algorithm provides a procedure to identify a region where all cells share a common property*. Using this procedure, different governing equations can be applied to different regions of a complex multiphase flow to reflect the different physics in these regions. For example, should one wish to treat the air bubbles trapped inside the water phase with compressible ideal gas law, she can apply Algorithm 1 to the air phase. All the air polygons in the air particle zone are clearly air bubbles. Relatively larger air bubbles are the 2-connected components (of the air material graph) enclosed by water–air interface cells.

Sometimes a certain region of a free-surface flow can be considered as a continuum of both air and water penetrating into each other, e.g., the region near a plunging wave-breaker. The topology of this region can be modeled by the material graph with (15) as the edge rule and the following as the vertex rule,

$$f_{i,j} \in [f_{\mathrm{low}}, f_{\mathrm{hi}}], \tag{17}$$

where $f_{i,j}$ is the volume fraction of water in cell $[i,j]$ and f_{low} and f_{hi} are two characteristic volume-fraction values of the air/water mixture. For example, Chanson [11] reported that for free-surface aeration in turbulent flows, the air/water mixture behaves as a homogeneous compressible fluid with continuous distribution of volume-fraction, velocity and bubble count rate for $0.1 < f_{i,j} < 0.7$. Thus, after applying Algorithm 1, a homogenization method, previously known as the WKB theory [6], can be used to approximate the continuum zone of the air/water mixture while the buffer zone serves as the boundary of the mixture. This approach enriches the methodology of the sharp interface formulation by treating the air/water mixture as a third phase, which is made possible by the single-phase decomposition algorithm.

Hereafter we treat the buffer zone and the continuum zone as a whole and refer to it as the 'continuum-buffer zone' or the *bulk* of the phase.

3. Mathematical formulation

3.1. Governing equations

The water phase is modeled as an incompressible inviscid fluid with the surface tension effect neglected. The governing equations in the continuum-buffer zone are thus the Euler equations:

$$\nabla \cdot \mathbf{u} = 0, \tag{18}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \mathbf{g} - \frac{\nabla p}{\rho}, \tag{19}$$

where density ρ is a material-wise constant. (18) and (19) are solved for the whole domain. Because of the reasons discussed in the next section, the air phase should not be taken as physically realistic air, but an 'extension' of the water phase.

For a water particle in the particle zone near the free-surface, its ambient pressure is that of the air phase. If we further neglect the interaction between water particles, the only dominant force is gravity. Therefore the governing equation in the particle zone describes the balance between gravity force and particle inertia,

$$\frac{D\mathbf{u}}{Dt} = \mathbf{g}. \tag{20}$$

For the time being, neglecting the interaction of particles excludes our model from applications to strongly fragmented free surfaces.

3.2. The passive-response assumption

One way to deal with the problem of physical jumps is the Immersed Interface Method (IIM) [42,43]. In this method, jump conditions are derived from physical requirements [44,92] and are explicitly incorporated into the finite difference schemes. However, for inviscid flows, the free-slip condition must be used at the free surface because of the absence of a viscous boundary layer. Even if the viscous boundary layer is present, the physically realistic free surface almost always has a contamination layer, which creates a boundary condition that is neither free-slip nor no-slip, due to the nonlinear response of the surfactants to flow-induced compression and expansion [89]. In both cases, the jump of the tangential velocity at the interface is not known *a priori* and explicit forms of the higher-order velocity-derivative jump conditions cannot be derived. Consequently, the IIM cannot be used in our model.

If the momentum variation of the air near the interface is small compared to that of the water, the anisotropic part of a stress exerted by the air on the water is negligible. This is often the case because of the large density ratio. Furthermore, neglecting surface tension and viscous effect leads to continuous pressure across the interface. More precisely, surfaces that are 'free' have negligible anisotropic stresses and cannot support variations in pressure. As a result, the free surface must move in such a way as to maintain the uniform pressure on the surface. It follows from the above argument that water is not influenced by air whereas air deforms freely in response to the change of the interface locus.

We thus assume that the velocity field does not have any jumps across the interface and the air phase responds *passively* to the water phase. In other words, we make the assumption that *physical information never propagates from the air into the water*. Although the computed air velocity deviates largely from that of the reality, this is acceptable if we do not care about the air velocity. The benefits are however of great convenience:

- Elimination of the velocity jumps yields simple algorithms since there is no need to impose velocity jump corrections.
- Continuous, divergence-free velocity field across the interface is conducive to interface tracking.

The latter is also the reason why the velocities in the air is solved even if we do not care about the air velocity. In VOF methods, the air is treated as a void where both velocity and pressure are set to zero. For newly-filled interface cells that were pure air cells in the previous time step, velocity has to be specified for interface tracking. Many VOF methods employ the 'momentum-capturing' method [12], which neglect spatial variations of the velocity field near the interface and equate the velocities in a newly-filled cell to those closest in the water. Indeed, if velocities in the air phase are not calculated, this is the only way to satisfy the divergence-free condition for conservative interface tracking. However, this crude treatment prevents convergence of interface tracking since it is essentially a zero-th order approximation of the velocity field and the

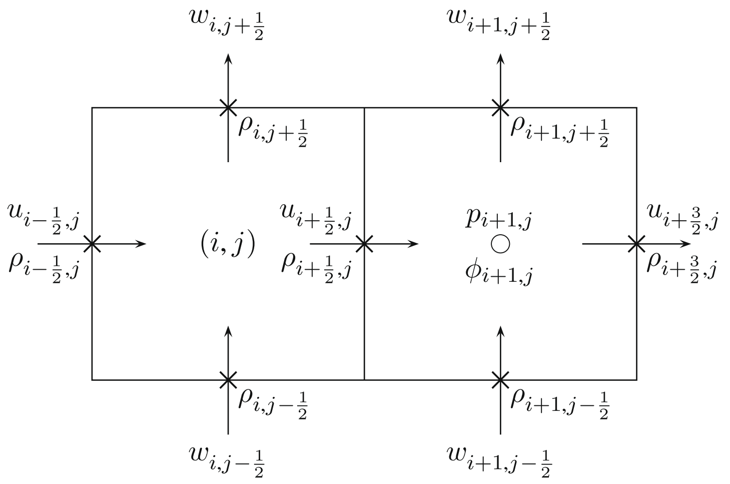


Fig. 3. Staggered arrangement. Pressure and its increment are located at cell centers ('○'); velocity and the averaged density are located at cell faces ('×').

interface is one-dimensional lower than the computational domain. Worse, the error of interface tracking further deteriorates the accuracy of the flow field calculation. To remedy this, we require that velocity in all the pure-air interface candidate cells be calculated under the passive response assumption. Thus, for a candidate cell that only contains air, the velocity should be considered as an *extension* of the water velocity, not the air velocity.

In Section 4.1, we show how a careful design of the numerical algorithm enforces the central idea of 'passive response'. It is important to point out that this 'passive-response' assumption does not hold for two-fluid flows with non-negligible interface stresses, such as wind-driven water waves. After all, technically these flows are not 'free-surface' flows in that the anisotropic stresses exerted by the air are non-negligible and substantially influence the movement of the interface.

4. Numerical implementation

4.1. A projection method tailored for free-surface flows

We use the following two-step projection method to solve the governing Eqs. (18) and (19) for the whole domain on a staggered rectangular mesh, where velocities are located at cell faces and pressure is located at cell centers, as shown in Fig. 3:

$$\frac{\mathbf{u}^* - \tilde{\mathbf{u}}^n}{\Delta t} = \mathbf{g} - \frac{\nabla p^n}{\rho} - (\tilde{\mathbf{u}}^n \cdot \nabla)\tilde{\mathbf{u}}^n, \tag{21a}$$

$$\nabla \cdot \left(\frac{\nabla \phi^{n+1}}{\rho}\right) = \frac{\nabla \cdot \mathbf{u}^*}{\Delta t}, \tag{21b}$$

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^*}{\Delta t} = -\frac{\nabla \phi^{n+1}}{\rho}, \tag{21c}$$

$$p^{n+1} = p^n + \phi^{n+1}, \tag{21d}$$

where $\mathbf{u}^n$ and p^n are known at the beginning of the time step. The pressure increment, ϕ, has a short lifetime of one step. The averaged ρ is at the same locations of velocities and has twice as many data points as those of the pressure. As shown in Fig. 3, $\rho_{i+\frac{1}{2},j}$ is averaged over the polygon with the four vertices located at $h(i,j-\frac{1}{2})^T$, $h(i+1,j-\frac{1}{2})^T$, $h(i+1,j+\frac{1}{2})^T$ and $h(i,j+\frac{1}{2})^T$. The current averaging in our method, done by intersecting this polygon with the water SSP in cells (i,j) and $(i+1,j)$, is much more accurate than that used in VOF methods, which stores cell-averaged densities at cell centers and calculates $\rho_{i+\frac{1}{2},j}$ by $\rho_{i+\frac{1}{2},j} = (\rho_{i,j} + \rho_{i+1,j})/2$.

$\tilde{\mathbf{u}}^n$ is defined as an auxiliary velocity field that differs from $\mathbf{u}^n$ only in interface candidate cells where $\tilde{\mathbf{u}}^n(\mathbf{x})$ is obtained by linear extrapolation from inside the water phase. Since the interface has a lower dimension than that of the domain, this extrapolation will not affect convergence of the velocity field calculation. Also, there always exists enough velocities in the water bulk since it is 2-connected.

In the first predictor step, an intermediate velocity $\mathbf{u}^*$ is calculated. Since the pressure gradient is evaluated in the RHS of (21a), $\mathbf{u}^*$ is a good approximation of $\mathbf{u}^{n+1}$ with an error of $O(\Delta t^2)$. The corrector step (21c) furthermore renders $\mathbf{u}^{n+1}$ divergence-free. Instead of solving for pressure as in Chorin's original projection method [13,14], our method solves for pressure increment and is in the category of the 'pressure-incremental' methods [5].

The enforcement of the 'passive-response' of the air phase concerns the discretization of the advection terms and the solution of the Poisson Eq. (21b), which are discussed in the following sections.

4.1.1. Discretization of the advection terms

Hereafter let Ω^- and Ω^+ be the water and air phases, respectively; (x_0, y_0) be the origin of the computational domain, the location vector $\mathbf{x}_{i,j} = (x_0 + i\Delta x, y_0 + j\Delta y)$, and the velocity vector $\mathbf{u} = (u, w)$.

Referring to Fig. 3, $\tilde{u}\frac{\partial \tilde{u}}{\partial x}$ at the right face of cell $[i,j]$ is given by

$$\left(\tilde{u}\frac{\partial \tilde{u}}{\partial x}\right)_{i+\frac{1}{2},j} = u_{i+\frac{1}{2},j}\left\{\frac{1+\beta_{uu}}{2}\left(\frac{\partial u}{\partial x}\right)_{i,j} + \frac{1-\beta_{uu}}{2}\left(\frac{\partial u}{\partial x}\right)_{i+1,j}\right\}, \tag{22}$$

where

$$\beta_{uu} = \begin{cases} -1 & \mathbf{x}_{i+\frac{1}{2},j}, \mathbf{x}_{i+\frac{3}{2},j} \in \Omega^-; \quad \mathbf{x}_{i-\frac{1}{2},j} \in \Omega^+, \\ 1 & \mathbf{x}_{i+\frac{1}{2},j}, \mathbf{x}_{i-\frac{1}{2},j} \in \Omega^-; \quad \mathbf{x}_{i+\frac{3}{2},j} \in \Omega^+, \\ \alpha \operatorname{sign}\left(u_{i+\frac{1}{2},j}\right) & \text{otherwise}, \end{cases} \tag{23}$$

velocity gradients at cell centers are evaluated as

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} = \frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{\Delta x}, \tag{24}$$

and α is a constant for tweaking between central difference ($\alpha = 0$) and the upwind scheme ($\alpha = 1.0$). We use $\alpha = 0.41$ for a balance between accuracy and stability.

If $\mathbf{x}_{i+\frac{1}{2},j}$, $\mathbf{x}_{i+\frac{3}{2},j}$, and $\mathbf{x}_{i-\frac{1}{2},j}$ are all inside the water phase, the third case of (23) is matched and (22) amounts to a mixture of central difference and upwind scheme. If $\mathbf{x}_{i-\frac{1}{2},j}$ is not inside the water phase, (22) simply reduces to

$$\left(\tilde{u}\frac{\partial \tilde{u}}{\partial x}\right)_{i+\frac{1}{2},j} = u_{i+\frac{1}{2},j}\frac{u_{i+\frac{3}{2},j} - u_{i+\frac{1}{2},j}}{\Delta x}. \tag{25}$$

In this way the air velocities are never used in evaluating the advection terms. In other words, the first and second cases of (23) indicate that the velocity derivatives in the water phase is *always* preferred for an interface cell, which enforces the 'passive-response' assumption by ensuring no propagation of the velocity information from the air phase into the water phase. Similar arguments apply to the other advection terms $\tilde{w}\frac{\partial \tilde{u}}{\partial y}$, $\tilde{u}\frac{\partial \tilde{w}}{\partial x}$, and $\tilde{w}\frac{\partial \tilde{w}}{\partial y}$.

Eqs. (22) and (23) couple the projection method to material areas. A straightforward implementation will check the cell type and material polygons before evaluating the schemes. This procedure slows down the program considerably. Fortunately, we can replace the air velocities in the interface cells by linearly extrapolating from inside the water phase, then (22) and (23) are equivalent to

$$\left(\tilde{u}\frac{\partial \tilde{u}}{\partial x}\right)_{i+\frac{1}{2},j} = \tilde{u}_{i+\frac{1}{2},j}\left\{\frac{1+\alpha\mathrm{sign}\left(\tilde{u}_{i+\frac{1}{2},j}\right)}{2}\left(\frac{\partial \tilde{u}}{\partial x}\right)_{i,j} + \frac{1-\alpha\mathrm{sign}(\tilde{u}_{i+\frac{1}{2},j})}{2}\left(\frac{\partial \tilde{u}}{\partial x}\right)_{i+1,j}\right\} \tag{26}$$

because the linear extrapolation essentially duplicates the velocity gradient at the water side to that of the air side.

Since $\mathbf{u}^n$ is used for interface tracking after the projection steps, it cannot be replaced by $\tilde{\mathbf{u}}^n$. Thus, the extrapolation to obtain $\tilde{\mathbf{u}}^n$ requires additional temporary storage in computer memory. However, using (26) together with extrapolation has several advantages over a direct implementation of (22) and (23):

- *Simplified algorithm.* The coefficients β_{uu}, β_{uw}, β_{wu} and β_{ww} all have different forms with respect to the location vectors due to the staggered arrangement; using extrapolation avoids this complication.
- *Better performance.* Constantly checking whether a location is within the water phase slows down the evaluation of $\mathbf{u}^*$ and prevents optimization; after all, the extrapolation is of $O(1/h)$ due to the lower dimension of an interface while the direct implementation incurs an $O(1/h^2)$ computation.
- *Decoupled modules.* Introducing $\tilde{\mathbf{u}}$ enables us to separate the extrapolation algorithm from the projection steps and thus yields program modules with much less coupling.

Following Lemos [41], a 'balancing tensor viscosity'

$$(1-\alpha)\frac{\Delta t}{2}(\tilde{\mathbf{u}}\cdot\tilde{\mathbf{u}})\nabla^2\tilde{\mathbf{u}}$$

is added to the RHS of (21a) to correct the most destabilizing truncation error arising from the use of forward time differences. The factor $1-\alpha$ ensures that the amount of balancing tensor viscosity equals the amount of central differencing used in (26). Here again the extrapolation approach is very convenient for evaluating the diffusion terms.

4.1.2. The Poisson equation

Eq. (21b) is an Poisson equation with discontinuous coefficient $1/\rho$. One can solve it by a special elliptic solver such as the Ghost Fluid method [49]. However, the pressure-incremental formulation allows a simpler approach. Using the chain rule, we can rewrite (21b) as

$$\nabla^2\phi^{n+1} = \rho\frac{\nabla\cdot\mathbf{u}^*}{\Delta t} - \rho\left[\frac{1}{\rho}\right]\delta(\mathbf{x}-\mathbf{X})\mathbf{n}\cdot\nabla\phi^{n+1}, \tag{27}$$

where $\delta(\mathbf{x}-\mathbf{X})$ is the two-dimensional Dirac delta function, $\mathbf{X}$ represents the locus of the free surface and $\mathbf{n}$ the local interface normal vector.

Since (21a) implies $\mathbf{u}^{n+1} - \mathbf{u}^* = O(\Delta t^2)$, it follows from (21c) that $\nabla\phi^{n+1} = O(\Delta t)$. The second RHS term in (27) is thus much smaller than the first RHS term. Hence the density jump $[\frac{1}{\rho}]$ has negligible effects on the solution of the Poisson equation, because of the 'pressure-incremental' formulation. Furthermore, (21c) implies that the velocity correction to the water phase is much smaller than that of the air phase, due to the large density ratio.

Note that the Poisson equation numerically solved is not (27), but (21b). The only purpose of introducing (27) is to argue that (21b) can be solved as if there were no density discontinuities, which only cause negligible errors since the second term is much smaller than the first term in 15. Thus no special treatment is needed for the density jump and we use a semi-coarsening multigrid method [70] for solving (21b).

4.1.3. Boundary conditions

At the heart of any projection method it is the Helmholtz–Hodge decomposition [63], which states that a smooth vector field $\mathbf{u}^*$ in a bounded domain Ω can always be decomposed into a zero-curl part $\nabla\phi$ and a zero-divergence part $\mathbf{u}$:

$$\mathbf{u}^* = \mathbf{u} + \nabla\phi, \tag{28}$$

where

$$\nabla \bullet \mathbf{u} = 0, \tag{29a}$$

$$\oint_{\partial\Omega} (\mathbf{u} \bullet \mathbf{n}) = 0 \tag{29b}$$

and $\mathbf{n}$ is the unit outward normal vector of $\partial\Omega$.

In fact, the divergence of (28)

$$\nabla^2\phi = \nabla \bullet \mathbf{u}^* \tag{30}$$

and the normal component of (28)

$$\frac{\partial\phi}{\partial n} = \mathbf{u}^* \bullet \mathbf{n} - \mathbf{u} \bullet \mathbf{n} \tag{31}$$

defines an elliptic problem for ϕ with Neumann boundary conditions, which has a unique solution, up to an additive constant, provided that the solvability condition

$$\int_{\Omega} \nabla \bullet \mathbf{u}^* = \oint_{\partial\Omega} \frac{\partial\phi}{\partial n} = \oint_{\partial\Omega} (\mathbf{u}^* \bullet \mathbf{n} - \mathbf{u} \bullet \mathbf{n}) \tag{32}$$

is satisfied. Eq. (32) always holds because of (29b) and the divergence theorem.

As noted in [9], the boundary conditions of a projection method must comply to (28) and the solvability condition (29b).

Since $\nabla\phi = O(\Delta t)$ (see Section 4.1.2), we can set $\frac{\partial\phi}{\partial n} = 0$ and $\mathbf{u}^*|_{\partial\Omega} = \mathbf{u}^{n+1}|_{\partial\Omega}$. Note that this is another advantage of the pressure-incremental formulation. Suppose the pressure gradient term were left out in (21a) and pressure instead of its increment were solved for in (21b), the boundary condition for u_* would have been non-trivial in that (31) would have to be used since using $\frac{\partial\phi}{\partial n} = 0$ violates (28), resulting in a non-convergent scheme [36].

For consistency with the passive response assumption, we always use open boundary conditions for air velocity.

4.2. Computation cycle

In addition to the SSP data structure for the PAM method, we also maintain a list of particles that records their masses and velocities. Given velocity $\mathbf{u}^n$ and pressure p^n in the whole domain, the particle list and the material SSP $\mathcal{M}^n$ of the water phase, HyPAM updates the particle list and calculates $\mathbf{u}^{n+1}$, p^{n+1}, and water SSP $\mathcal{M}^{n+1}$ by the following steps:

(i) copy $\mathbf{u}^n$ to $\tilde{\mathbf{u}}$ and extrapolate $\tilde{\mathbf{u}}$ for interface candidate cells,
(ii) calculate $\mathbf{u}^{n+1}$ and p^{n+1} using (21),
(iii) with $\mathbf{u}^n$, $\mathbf{u}^{n+1}$ and a linear interpolation in time, advance $\mathcal{M}^n_{i,j}$ to $\mathcal{M}^{n+1}_{i,j}$ using the PAM method,
(iv) update the positions and velocities of the particles by (20),
(v) check whether any particle has traveled into a material polygon $\mathcal{M}^{n+1}_{i,j}$ in the continuum zone. If so, remove that particle from the list and increase the volume of $\mathcal{M}^{n+1}_{i,j}$ by the mass of the particle using the volume-adjusting algorithms in [96],
(vi) call Algorithm 1 on $\mathcal{M}^n$ to decompose the water phase into a continuum-buffer zone and a particle zone,
(vii) for each cell $[i,j]$ in the particle zone, remove $\mathcal{M}^n_{i,j}$ from $\mathcal{M}^n$, change it into a point mass and add the point mass into the particle list with its velocity interpolated from $\mathbf{u}^{n+1}$.

In each time step, the coupling of the continuum zone and the particle zone is considered in two aspects. On the one hand, we check whether the water phase contains a particle zone. If so, we put these particles into the particle list. On the other hand, we check whether any water particles have traveled into the water bulk. If so, we take them off the particle list and enforce the mass conservation in the interface cell under collision. Since the length-scale of a particle is never bigger than the grid size, we assume that the particle undertakes the velocity in the continuum zone at the collision site. Due to this treatment, the whole system is dissipative.

As for the velocity field in interface tracking, we use $\mathbf{u}^n$ and $\mathbf{u}^{n+1}$ instead of $\tilde{\mathbf{u}}^n$ and $\tilde{\mathbf{u}}^{n+1}$ because the former ones are divergence-free. This seemingly trivial subtlety has a significance on the accuracy of interface tracking, since the PAM method does not change the velocity field near the interface as some VOF methods do.

5. Tests

Forced-flow tests are a common technique to examine the convergence rate of a flow solver. In this technique, analytical forms for velocity and pressure are first designed, usually with the velocity formula satisfying the divergence-free condition; then the gravitational body force is replaced by another driving force to close the momentum equation.

5.1. Single-phase forced-flows

As preliminary tests, two forced-flow tests are performed to verify the consistency of the projection method of HyPAM for a *single* phase.

The first test [36] has the following flow field

$$u(x,y,t) = -e^{-2t}\cos x \sin y, \tag{33a}$$

$$w(x,y,t) = e^{-2t}\sin x \cos y, \tag{33b}$$

$$p(x,y,t) = -\frac{1}{4}(\cos(2x)+\cos(2y))e^{-4t}. \tag{33c}$$

The flow field in the second test [9] is strongly transient:

$$\omega(t) = 1+\sin(2\pi t^2), \tag{34a}$$

$$\theta = 2\pi(x-\omega(t)), \tag{34b}$$

$$u = \cos\theta(3y^2-2y), \tag{34c}$$

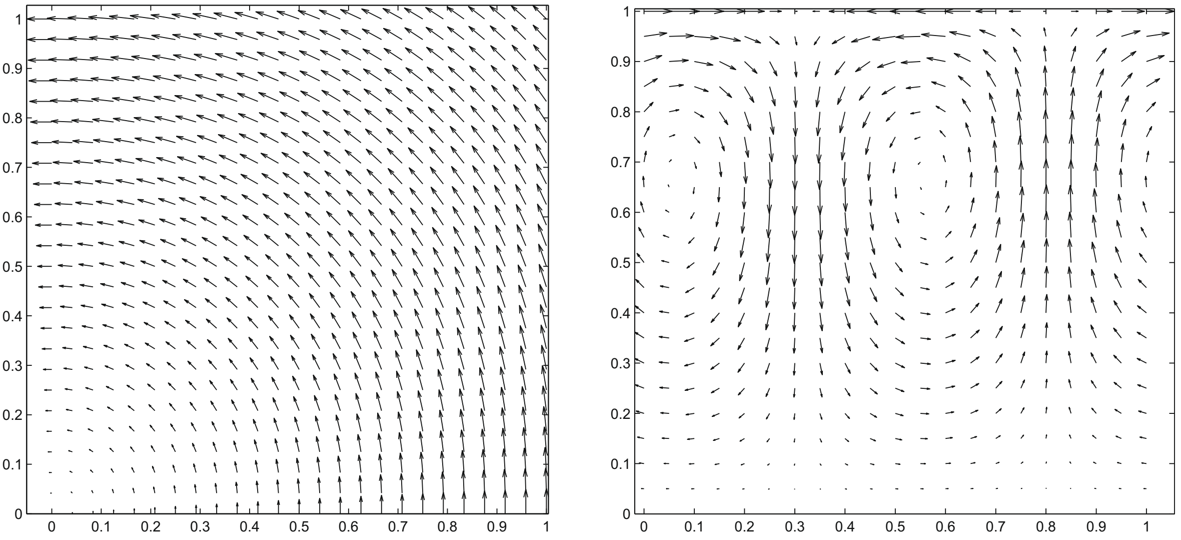
$$w = 2\pi \sin\theta y^2(y-1), \tag{34d}$$

$$p = -\frac{\omega'(t)}{2\pi}\sin\theta(\sin(2\pi y)-2\pi y+\pi) + v\cos\theta(\sin(2\pi y)-2\pi y+\pi), \tag{34e}$$

where the kinematic viscosity v is set to zero.

In both of the tests, explicit forms of the driving body forces can be derived from (18) and (19). In a unit square domain of [0, 1] ×[0, 1], we use the exact solutions of $\mathbf{u}$ and p to set the initial conditions, then $\mathbf{u}$ and p are advanced to $T = 0.1$ using (21) on five successively refined grids. In Fig. 4, we show the velocity fields at T based on the results of the finest grids. First-order convergence with respect to 1-norm and ∞-norm is evident in Table 2 for (34) and in Table 1 for (33).

Since there are no interface cells and $\tilde{\mathbf{u}} = \mathbf{u}$, these forced-flow tests verify the consistency of the projection method (21) for single phase flows.



(a) forced flow (33) with $h = 1/192$ (b) forced flow (34) with $h = 1/160$

Fig. 4. Velocity results of HyPAM for the forced flow tests at $T = 0.1$.

Table 1
Convergence of HyPAM on the forced flow (33) (Neumann BC).

h	1/12	1/24	1/48	1/96	1/192
$\Vert p^{T/\Delta t} - p(T)\Vert_\infty$	8.27e−03	2.40e−03	8.54e−04	4.15e−04	2.11e−04
$\Vert p^{T/\Delta t} - p(T)\Vert_1$	6.89e−03	2.05e−03	6.96e−04	2.69e−04	1.15e−04
$\Vert \mathbf{u}^{T/\Delta t} - \mathbf{u}(T)\Vert_\infty$	3.40e−02	2.35e−02	1.32e−02	6.65e−03	3.21e−03
$\Vert \mathbf{u}^{T/\Delta t} - \mathbf{u}(T)\Vert_1$	1.27e−02	9.15e−03	5.33e−03	2.62e−03	1.20e−03

Table 2
Convergence of HyPAM on the forced flow (34) (Dirichlet BC).

h	1/10	1/20	1/40	1/80	1/160
$\|p^{T/\Delta t} - p(T)\|_\infty$	6.66e−02	2.76e−02	1.16e−02	5.33e−03	2.56e−03
$\|p^{T/\Delta t} - p(T)\|_1$	3.44e−02	1.38e−02	5.98e−03	2.76e−03	1.32e−03
$\|\mathbf{u}^{T/\Delta t} - \mathbf{u}(T)\|_\infty$	9.84e−02	4.64e−02	2.26e−02	1.12e−02	5.71e−03
$\|\mathbf{u}^{T/\Delta t} - \mathbf{u}(T)\|_1$	3.52e−02	1.77e−02	8.91e−03	4.47e−03	2.24e−03

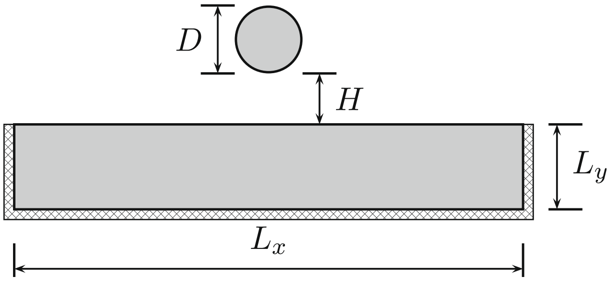


Fig. 5. Numerical setup of a droplet impact test. $D = 0.24$ m, $H = 0.1$ m, $L_x = 3.0$ m and $L_y = 0.5$ m.

5.2. Droplet impact

We simulate a droplet impact problem to illustrate the main features of HyPAM. The setup of the problem is shown in Fig. 5, where the depth and the length of the reservoir are $L_y = 0.5$ m and $L_x = 3.0$ m, respectively. The circular water droplet with a diameter $D = 0.24$ m is initially at a height $H = 0.1$ m away from the reservoir. The velocity of the droplet at the time of impact, $u_* = \sqrt{2gH}$, characterizes the free surface disturbance.

For this setup, the Reynolds number is $Re = u_* D/\nu = D\sqrt{2gH}/\nu \approx 3.4 \times 10^5$, the Froude number is $Fr = u_*^2/gD = 2H/D \approx 0.83$, and the Weber number is $We = \rho u_*^2 D/\sigma = (D/L_*^\sigma)^2 \approx 7.9 \times 10^3$. The large diameter of the droplet results in large Reynolds number and Weber number, which confirms the applicability of our model to this setup. Nonetheless, the main purpose of this test is not to capture the physical processes, but to illustrate the features of HyPAM, such as the separation of water particles from the water bulk, the merging of water particles back into the water bulk, numerical convergence and the conservation of mass and energy.

5.2.1. The hybrid features of HyPAM

Figs. 6 and 7 show five snapshots of water bulk velocities and water polygons in interface candidate cells. In Fig. 6(a), the water droplet is just about to impact on the reservoir. The circular shape of the droplet is well preserved, since the velocity extrapolation from inside the water phase ensures the uniformness of the velocity near the interface. This is also a confirmation of passive-response of the air phase since interface tracking and velocity field are coupled. Also, the time of this frame ($t = 0.135$ s) being close to the expected impact time ($t = 0.141$ s) indicates that the free fall of the water droplet is not influenced by air. This implies a good enforcement of the passive-response assumption.

In Fig. 6(b), the droplet merges into the reservoir and traps some air bubbles, which are implicitly defined by the enclosing water polygons. Although this feature mimics the local aeration mechanism [11], the bubbles should not be taken seriously because a two-dimensional droplet is far from the three-dimensional reality. However, it should be pointed out that the air bubbles in a HyPAM simulation are much closer to reality than those in VOF methods because of two reasons:

- Both PAM and VOF methods define a small air volume fraction f_{tiny}, below which the cell is considered as a pure-water cell. Thus an air bubble will be consumed during its advection. In VOF methods $f_{\text{tiny}} \approx 0.01$–$0.05$ because setting $f_{\text{tiny}} < 0.01$ creates spurious water debris near the interface. In contrast, f_{tiny} in HyPAM is limited only by the machine precision and can be as small as 10^{-9}. We set $f_{\text{tiny}} = 10^{-4}$ for all HyPAM tests in this paper;
- For a bubble smaller than $3h$, VOF methods cannot determine their location accurately because of their intrinsic limitation of the reconstruction step, which depends heavily on taking gradient of volume fractions. In contrast, HyPAM is able to track small air bubbles accurately, thanks to the polygonal representation of material area. This leads to a great advantage of HyPAM over VOF methods that HyPAM decouples the resolution of interface tracking from that of a flow solver. Because of this, bubbles smaller than $3h$ are tracked much more accurately; one big air bubble can even split into two smaller ones so long as the velocity field has such a tendency.

Since the Euler equation does not have a viscous term, the velocity field has a discontinuity along the tangential direction of the merging edge. In the numerical simulation, this discontinuity is smoothed out by spatial numerical diffusions, which are leading-order truncation terms of a first-order method.

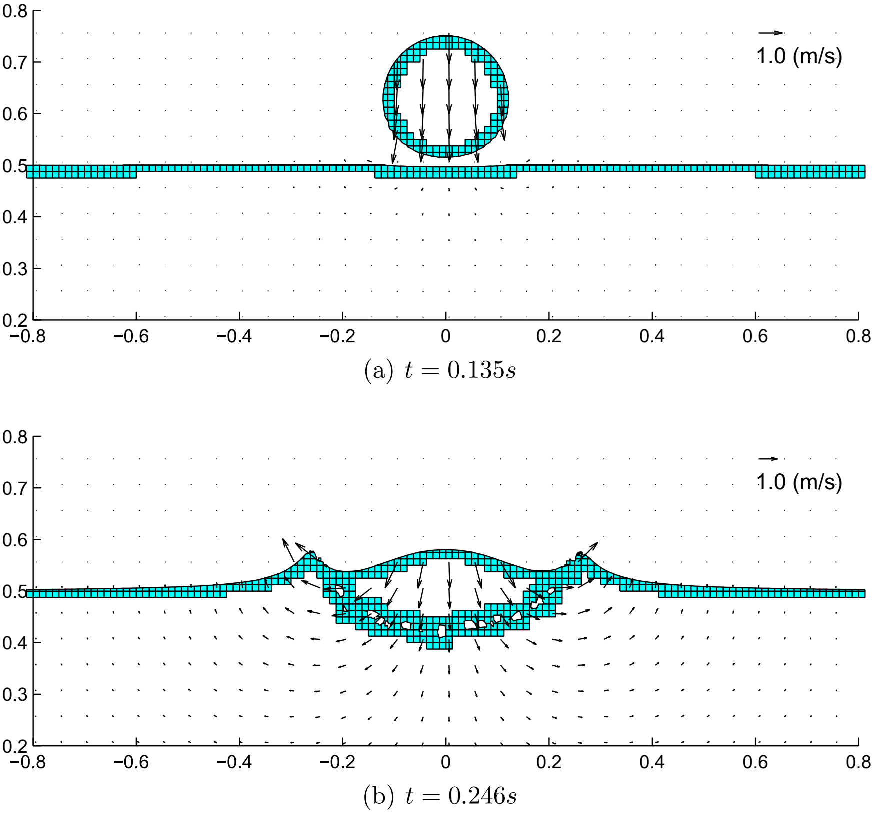


Fig. 6. Water velocity and water polygons near the interface of a droplet impact test ($t = 0.135$ s,0.246 s). The axes are spatial coordinates in meters. $\Delta x = \Delta y = h = D/20$. Material polygons of the interface candidate cells are shown on actual grids and one velocity arrow is drawn for every 4-by-4 cells. Water particles are shown in small isolated squares.

In Fig. 7(a), water droplets form near the two peaks of the free surface, due to the complex interface topology change. Since Algorithm 1 is performed at each time step, there can be an accumulation of water particles over a number of time steps. While some particles have reached their highest points, others have just begun the free fall. In Fig. 7(b), all the water particles are traveling downwards and merging back to the water bulk. Fig. 7(c) shows the uprise of the water in the reservoir as an aftermath of the droplet impact. There is no particle zone for the water phase in this frame.

The results in Figs. 6 and 7 are not exactly symmetric. This is due to the following reasons:

- The circular shape was initially approximated by a polygon that is not symmetric with respect to the axis $x = 0$.
- In the PAM method, the material polygons are adjusted in a way [96] to enforce mass conservation and ensure stability, but symmetry is not guaranteed.
- Although Neumann boundary conditions are used to solve (21b), it requires one Dirichlet boundary condition as the datum of ϕ. In this simulation, $\phi = 0$ is set to the up-left corner of the domain so as to minimize its disturbance to the solution of Euler equations. Although only $\nabla\phi$ is used in the corrector step, the asymmetry of ϕ is accumulated into the pressure field and ∇p results in a slightly asymmetric velocity field.

5.2.2. Mass and energy conservations

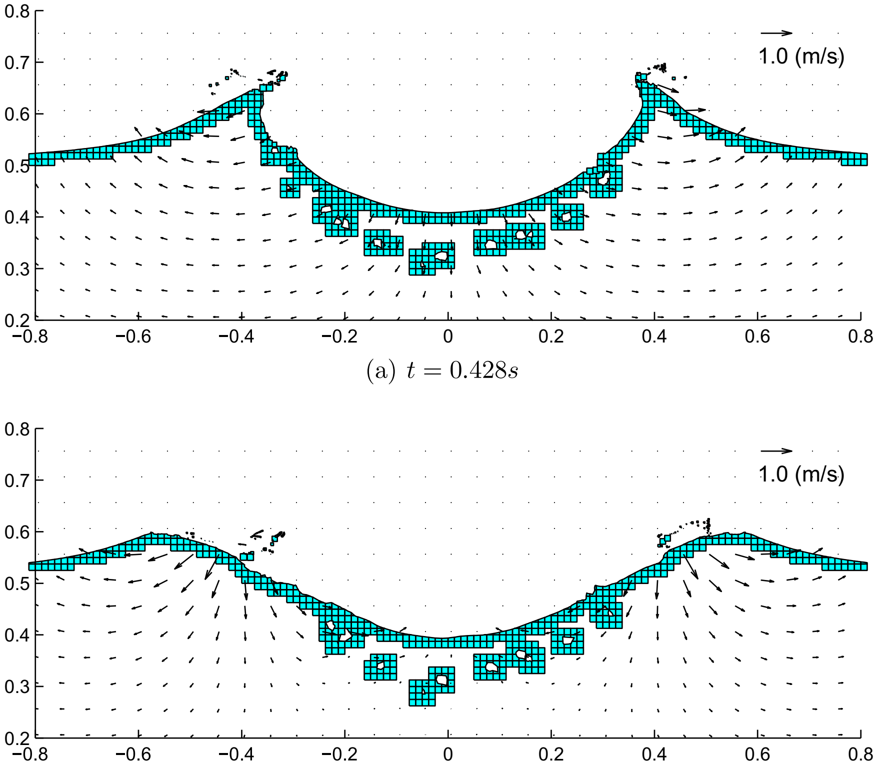
Let $\|\mathcal{M}(t)\|$ be the total mass of the water phase, including both the continuum-buffer zone and the particle zone, The mass considered for conservation is

$$M_d(t) = \|\mathcal{M}(t)\| - M_{\text{reservoir}}, \tag{35}$$

where $M_{\text{reservoir}} = L_x L_y$ is the total mass of the reservoir at $t = 0$. We show the evolution of $M_d(t)/M_d(0)$ on three successively refined grids in Fig. 8(a) and the corresponding error in the second row of Table 3. Mass conservation converges very fast as the grid size is reduced.

Define the total kinetic energy as

$$E_k(t) = \frac{h^2}{2}\sum_{i,j}\rho_{i,j}(t)(u_{i,j}^2(t) + w_{i,j}^2(t)) + \sum_{\ell}\frac{m_\ell}{2}(u_\ell^2(t) + w_\ell^2(t)), \tag{36}$$



(a) $t = 0.428s$

(b

where the summation i,j is taken over the continuum-buffer zone with respect to cell indices, $\rho_{i,j}(t)$ and $u_{i,j}(t), w_{i,j}(t)$ are the averaged density and velocities, respectively, at the cell centers; the summation ℓ is taken over the particle zone for all water particles, m_ℓ and $u_\ell(t), w_\ell(t)$ are the mass and velocities, respectively, of the ℓth particle.

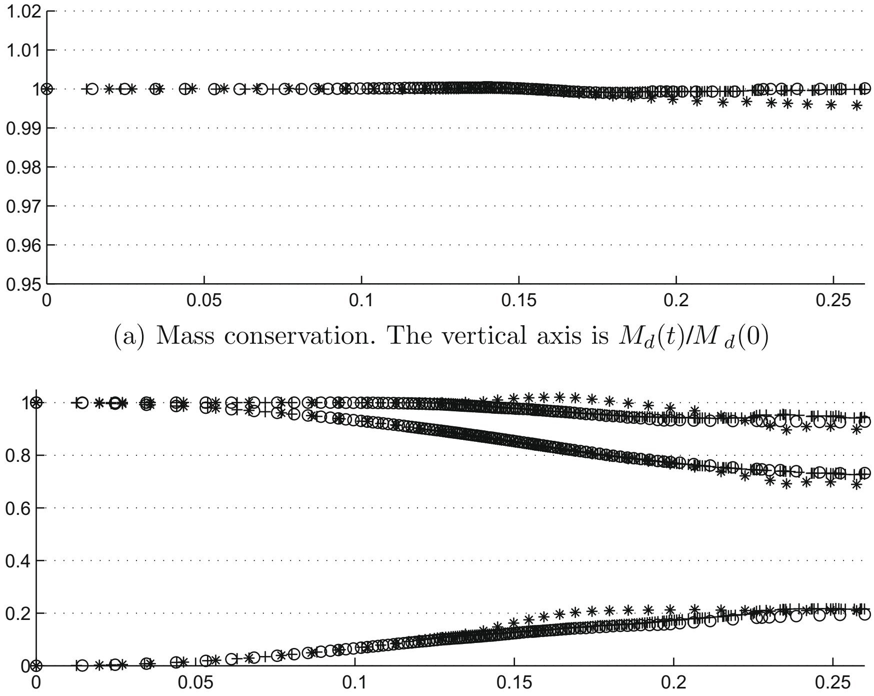
Similarly, the total potential energy is defined as

$$E_p(t) = h^2 \sum_{i,j} \rho_{i,j}(t) g y_j + \sum_{\ell} m_\ell g y_\ell - E_{\text{reservoir}}, \tag{37}$$

where y_j is the height of the cell center, and the potential energy of the reservoir is $E_{\text{reservoir}} = \rho_w g L_x L_y^2/2$ with ρ_w being the density of water. The total energy is then

$$E_T(t) = E_k(t) + E_p(t). \tag{38}$$

Since the initial potential energy of the reservoir is not considered in the energy budget and the initial condition of velocity is zero everywhere, the total energy at $t = 0$ is $E_T(0) = E_p(0)$, i.e. the energy budgets are examined only for the circular droplet.



(a) Mass conservation. The vertical axis is $M_d(t)/M_d(0)$

(b) Energy conservation. The highest lines are $E_T(t)/E_T(0)$, the middle lines $E_p(t)/E_T(0)$ and the lowest lines $E_k(t)/E_T(0)$.

Fig. 8. Mass and energy conservation of droplet impact tests. The horizontal axis is time in seconds. ‘**’: $h = D/10$, one ‘*’ represents two actual time steps; ‘∘∘’: $h = D/20$, one ‘∘’ represents four actual time steps; ‘++’: $h = D/40$, one ‘+’ represents eight actual time steps. The Courant number $C_r = 0.4$ for all cases.

Table 3
Conservation error of droplet impact tests. $t_e = 0.26s$ is the instant when the droplet has completely merged into the reservoir.

$\Delta x = \Delta y = h$	$D/10$	$D/20$	$D/40$
$\lvert \frac{M_d(t_e)}{M_d(0)} - 1 \rvert$	4.0e−3	2.0e−4	<1.0e−6
$\lvert \frac{E_T(t_e)}{E_T(0)} - 1 \rvert$	9.0e−2	7.5e−2	5.6e−2

The time history of energy budgets is shown in Fig. 8(b) and the corresponding error is shown in the third row of Table 3. Before the impact at $t = 0.141$ s, the potential energy of water droplet is converted to kinetic energy by gravity, without energy loss in the free fall. On the coarsest mesh, the total energy loss due to the droplet impact is 9%, since there are only 10×10 grids resolving the droplet. As the mesh is refined, the total energy loss of the impact is reduced to about 5% on the finest mesh. As noted in Section 4.2, the way of merging a water particle back into the continuum zone makes the system dissipative, thus in general energy is not conserved. However, the time instant of energy conservation examination is carefully chosen such that no water particles have been generated yet. Therefore, we expect energy conservation in the limit of $h \to 0$, although Fig. 8(b) and Table 3 suggest a much slower convergence rate than that of mass conservation.

5.2.3. Convergence

To demonstrate convergence, we run the droplet test on four successively refined grids with $h = D/N$ and $N = 5, 10, 20, 40$. The error of velocity and pressure on grids with $h = D/N$ is defined as

$$\epsilon_q^N = \frac{\sum_{i,j} |q^N(T) - q^{2N \to N}(T)|}{n_W}, \tag{39}$$

where the superscript N represents the $h = D/N$ grids and $T = 0.25$ s the instant when the droplet has just merged into the reservoir with no water particles having been generated yet. q, the physical quantity under examination, can be p for pressure or u, w for velocities. The superscript $2N \to N$ stands for a fictitious grids with $h = D/N$, but with its data sampled from those of the $D/2N$ grids by a two-dimensional tent filter. A cell in this fictitious grids is a pure cell if and only if all the four corresponding smaller cells in the $D/2N$ grids are pure cells of the same type; otherwise it is an interface cell. A cell index is excluded from the summation if and only if it denotes pure air cells on both the fictitious grids and the D/N grids, because the physical quantities in the air phase are only 'extensions' of the water phase. In other words, the summation is taken over cells that contain water on at least one of the two grids; n_W is the total number of such cell indices. Note that large errors might result from the case that one cell index corresponds to two different cell types on the two grids.

The interface tracking errors have already been included in (39) implicitly. In the 1-norm sense, interface tracking has already converged. However, to examine it more explicitly, we define a more stringent error criteria by the following

$$\epsilon_f^N = \|\mathcal{M}_W^N(T) \wedge \mathcal{M}_W^{2N}(T)\|, \tag{40}$$

where $\mathcal{M}_W^N(T)$ is the SSP of the water phase on the D/N grids at time T, $\|\cdot\|$ yields the total volume of a SSP, and '$\wedge$' represents the exclusive-or operation on 2-D compact point sets.

The convergence rate is defined as

$$\mathcal{O}^N = \log_2 \frac{\epsilon^{N/2}}{\epsilon^N} \tag{41}$$

ϵ_p^N, ϵ_u^N and ϵ_w^N, the errors of pressure and velocity, are shown in Table 4 that clearly demonstrate convergence. The interface tracking errors ϵ_f^N show convergence rates smaller than one. There are two main reasons for this: (1) at the time when the droplet impacts on the reservoir, the convex hull approximation in the PAM method is repeatedly invoked, generating large errors. (2) The entrapped bubbles are implicitly defined by the water polygons; the generation of these bubbles is coupled to the value of f_{tiny}, the convex hull approximation and the volume-adjusting algorithms. In Section 5.3, we shall show that interface tracking in HyPAM is convergent so long as there are no air bubbles and the convex hull approximation is not repeatedly invoked.

The criteria of numerical connectedness relates closely to the grid size. On the one hand, a finer resolution tends to better resolve the topology of a single phase, resulting in a larger continuum zone; on the other hand, a finer resolution also tends to better resolve a nonlinear velocity field near the interface and produces more particles. Thus the total mass in the particle zone is not only a function of the grid size, but also a function of the specific type of the flow under study. In Fig. 9, we show the total particle mass evolutions on different grids. On the coarsest grid, particle generation appears to be intermittent while on the finest grid the particle mass evolution curve is much smoother. The coarsest and finest grids seem to agree on a 'characteristic' total particle mass $M_{\text{particle}} = 2 \times 10^{-3} (m^2)$, which is about 0.6% of the droplet mass. It is speculated that the values of M_{particle} vary with different flow regimes, i.e., $M_{\text{particle}} = M(Re, Fr, We)$. It is important to point out that, as the grid size is reduced, HyPAM tends to produce more particles than it should since surface tension is neglected. This is probably the reason that no convergence is observed on the particle mass evolution.

We point out that the grid size for HyPAM should be determined by the velocity field rather than interface tracking. This is not true for many VOF-based continuum methods, since one often has to refine the mesh for the sole purpose of improving

Table 4
Convergence of the droplet test.

	$N = 5$	$N = 10$	$\mathcal{O}^{10}$	$N = 20$	$\mathcal{O}^{20}$
ϵ_p^N	1.07e+2	0.40+2	1.21	0.13e+2	1.87
ϵ_u^N	3.51e−2	0.91e−2	1.95	0.44e−2	1.05
ϵ_w^N	2.41e−2	1.00e−2	1.28	0.39e−2	1.35
ϵ_f^N	7.14e−2	5.41e−2	0.40	3.69e−2	0.55

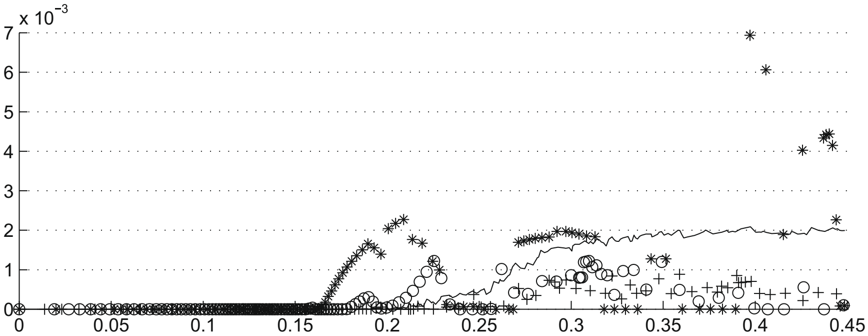


Fig. 9. Evolution of total mass in the particle zone for the droplet impact test. x-axis is the simulation time in seconds and y-axis is the total mass in the particle zone in m^2. '$*$': $h = D/5$; '$\circ$': $h = D/10$; '+': $h = D/20$; line: $h = D/40$.

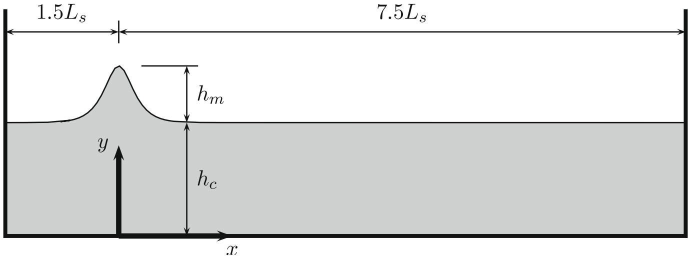


Fig. 10. Setup of solitary wave propagation tests. $h_c = 1.0$ m, $r_h = h_m/h_c = 0.5$, $L_s = 11.37h_c$.

interface tracking. The efficiency of HyPAM is thus better than that of VOF-based continuum methods because a $O(1/h^{d-1})$ phenomenon does not incur a $O(1/h^d)$ refinement.

5.3. Solitary wave propagation

In this test a solitary wave propagating in a constant water depth h_c is simulated. As shown in Fig. 10, the solitary wave is characterized by the maximum wave height h_m, the wave length L_s, and the time scale $T_* = L_s/\sqrt{gh_c}$. L_s is chosen to be the distance between the two locations where the surface displacement $\zeta' = h_c\zeta$ is one percent of h_m. In addition, two dimensionless variables are defined:

$$r_h = \frac{h_m}{h_c}, \tag{42}$$

$$r_L = \frac{h_c}{L_s}. \tag{43}$$

The spatial coordinates and time can be nondimensionalized by

$$\bar{x} = \frac{x}{L_s}, \quad \bar{y} = \frac{y}{h_c}, \quad \bar{t} = \frac{t}{T_*}. \tag{44}$$

For inviscid, irrotational free-surface flows, Grimshaw [30] presented an analytical solution for the free-surface displacement up to $O(r_h^2)$ as the following:

$$\zeta = s^2 - \frac{3}{4}r_h s^2 q^2 + r_h^2 s^2 q^2\left(\frac{5}{8} - \frac{101}{80}s^2\right), \tag{45}$$

where

$$\begin{aligned}
s &= \operatorname{sech}\left[\frac{\beta}{r_L}(\bar{x} - C\bar{t})\right],\\
q &= \tanh\left[\frac{\beta}{r_L}(\bar{x} - C\bar{t})\right],\\
\beta &= \frac{\sqrt{3r_h}}{2}\left(1 - \frac{5}{8}r_h + \frac{71}{128}r_h^2\right),\\
C &= \sqrt{1 + r_h - \frac{1}{20}r_h^2 - \frac{3}{70}r_h^3}.
\end{aligned}$$

The corresponding velocity field, nondimensionalized as

$$\bar{u} = \frac{u}{r_h\sqrt{gh_c}}, \quad \bar{w} = \frac{r_L w}{r_h\sqrt{gh_c}}, \tag{46}$$

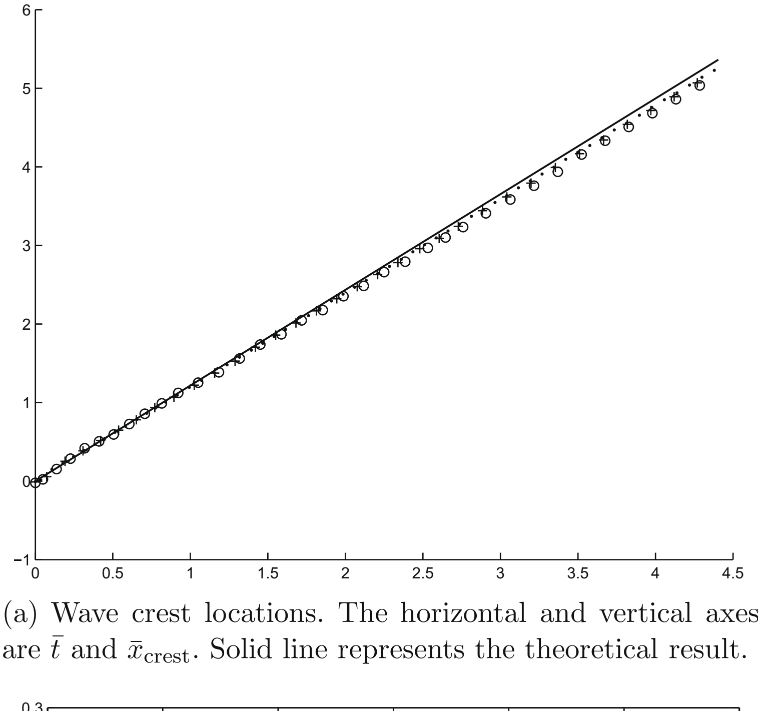
can be expressed as

$$\bar{u} = s^2\left\{1 + r_h\left[\frac{1}{4} - s^2 + \bar{y}^2\left(-\frac{3}{2} + \frac{9}{4}s^2\right)\right] + r_h^2\left[-\frac{19}{40} - \frac{1}{5}s^2 + \frac{6}{5}s^4 + \bar{y}^2\left(\frac{3}{2} + \frac{15}{4}s^2 - \frac{15}{2}s^4\right) + \bar{y}^4\left(\frac{3}{8} - \frac{45}{16}s^2 + \frac{45}{16}s^4\right)\right]\right\}, \tag{47}$$

$$\begin{aligned}
\bar{w}(\bar{x},\bar{y},\bar{t}) = s^2 q r_L \bar{y}\sqrt{3r_h}\Bigg\{1 &+ r_h\left[-\frac{3}{8} - 2s^2 + \bar{y}^2\left(-\frac{1}{2} + \frac{3}{2}s^2\right)\right] + r_h^2\Bigg[-\frac{49}{640} + \frac{17}{20}s^2 + \frac{18}{5}s^4 + \bar{y}^2\left(\frac{13}{16} + \frac{25}{16}s^2 - \frac{15}{2}s^4\right)\\
&+ \bar{y}^4\left(\frac{3}{40} - \frac{9}{8}s^2 + \frac{27}{16}s^4\right)\Bigg]\Bigg\}.
\end{aligned} \tag{48}$$

Numerical simulations are performed on three successively refined non-uniform grids with $\Delta x/5 = \Delta y = h_m/N$, where $N = 5, 10, 20$. (45), (47), (48) and the quasi hydro-static pressure are used for the initial conditions and free-slip walls are used for the boundary conditions. To avoid reflection, a simulation stops when the wave crest x_{crest} has just passed $x = 6L_s$.

Because of the constant water depth, theoretically the wave celerity should remain a constant. However, the solitary waves in the numerical results tends to travel slower as time progresses. In Fig. 11(a), wave crest locations of all three grids clearly lag behind the theoretical result for large $\bar{t}$. In Fig. 11(b), wave oscillations are very strong for the coarsest grids in the first two wave lengths of propagation. As for the finest grids, the solitary wave seems to approach to a steady wave speed after propagating for three wave lengths. The initial oscillation might be caused by the inconsistency between the exact solutions of the Euler equations and the initial conditions.



(a) Wave crest locations. The horizontal and vertical axes are $\bar{t}$ and $\bar{x}_{\text{crest}}$. Solid line represents the theoretical result.

(b) Wave speed errors. The horizontal and vertical axes are $\bar{x}_{\text{crest}}$ and $\Delta\bar{x}_{\text{crest}}/\Delta t - C$.

Fig. 11. Wave speed comparison. '○', '+' and '·' represent $N = 5, 10, 20$, respectively.

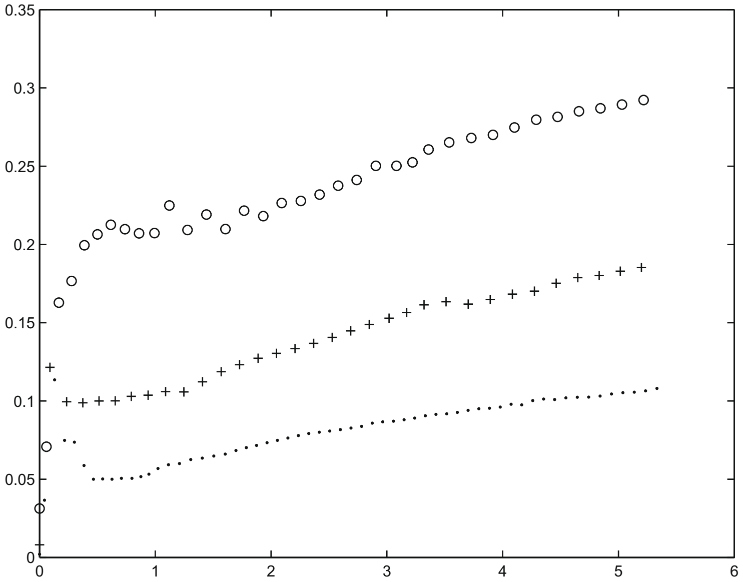


Fig. 12. Maximum wave height deficiency on different grids. The horizontal and vertical axes are $\bar{x}$ and $(h_m + h_c - y_{\text{crest}})/h_m$, respectively, with y_{crest} being the total water depth at the wave crest. '○', '+' and '·' represent $N = 5,10,20$, respectively.

Theoretically the maximum wave height should remain a constant throughout the propagation. In our simulations, numerical diffusion and dissipation tend to reduce it as the wave propagates. In Fig. 12, wave height deficiencies grow with the propagation distance. However, refining the grids reduces the wave height, with a convergence rate close to one.

To examine the convergence of the free surface displacement more carefully, the nondimensionalized wave forms at four instants are plotted against the exact solutions, as shown in Fig. 13. Correspondingly, the interface tracking error is then defined as the area difference between the numerical results and the exact solution:

$$\epsilon_{\zeta}^{N}(\bar{t}) = \int_{0}^{6} |\zeta^{N}(\bar{x},\bar{t}) - \zeta(\bar{x},\bar{t}))|\mathrm{d}\bar{x}, \tag{49}$$

where $\zeta^{N}(\bar{x},\bar{t})$ is the surface displacement on the h_m/N grids. This definition is equivalent to taking the exclusive-or operation between the exact water SSP and that of a numerical simulation.

In VOF methods, a large ratio of Δx to Δy often yield spurious interface reconstruction results, because interface normal is determined by taking gradient of the volume fraction values. In contrast, HyPAM yields good results even with a large ratio of spatial intervals. In addition, Table 5 clearly demonstrates first-order convergence on interface tracking. This confirms that HyPAM tracks the interface consistently so long as the convex hull approximation are not repeatedly invoked and no air bubbles are involved.

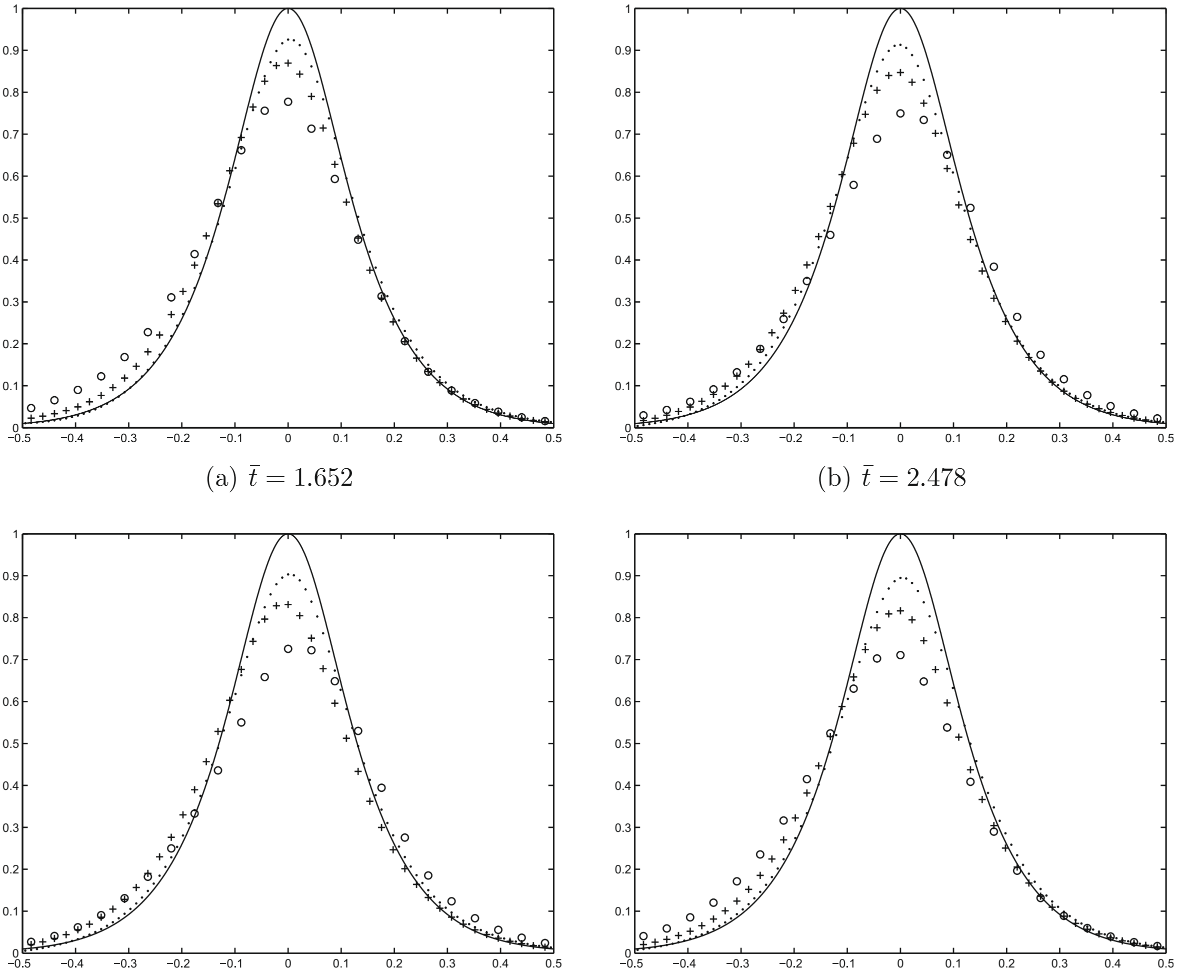
5.4. Dam-break

For a physical validation of HyPAM, we simulate a dam-break problem and compare the results to those of previous experiments [80] as well as the numerical results of COBRAS. This is also a necessary check before applying HyPAM to more complicated scenarios such as the swash zone hydrodynamics [97].

The numerical setup is shown in Fig. 14, where a gate at $x = 0$ separates two initially still reservoirs with different water depths. The upstream and downstream water surfaces meet the gate at point A and B, respectively. At time $t = 0$, the gate is removed instantaneously and the generated bore propagates on the downstream horizontal bed with a constant water depth 45% of that of the upstream reservoir. The length of the two reservoirs have been made long enough such that neither the forward-traveling waves nor the backward-traveling waves reaches the vertical boundaries in the examined time period. The upstream water depths $h_u = 0.1$ m and $h_u = 0.36$ m correspond to a small scale test and a large-scale test, respectively. Choosing the difference of the water depths as the length-scale, the Reynolds number and Weber number for the two cases are $Re = u_*L_*/\nu = 0.55h_u\sqrt{gh_u}/\nu \approx 5.44 \times 10^4, 3.72 \times 10^5$ and $We = \rho u_*^2 L_*/\sigma = 0.55(h_u/L_*^{\sigma})^2 \approx 754, 9778$, respectively.

5.4.1. Qualitative validation: the jet formation and spilling breakers

We show six snapshots of velocity field and free surface locations of the small-scale dam-break test in Fig. 15, where the first four frames are in the initial stage. Velocities in pure air cells are removed since they are non-physical. The initial pressure is hydrostatic and discontinuous across the gate. Since the fluids are incompressible, the sudden removal of the gate



(a) $\bar{t} = 1.652$

(b) $\bar{t} = 2.478$

(c) $\bar{t} = 3.304$

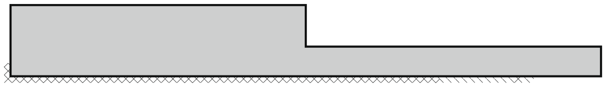
(d) $\bar{t} = 4.130$

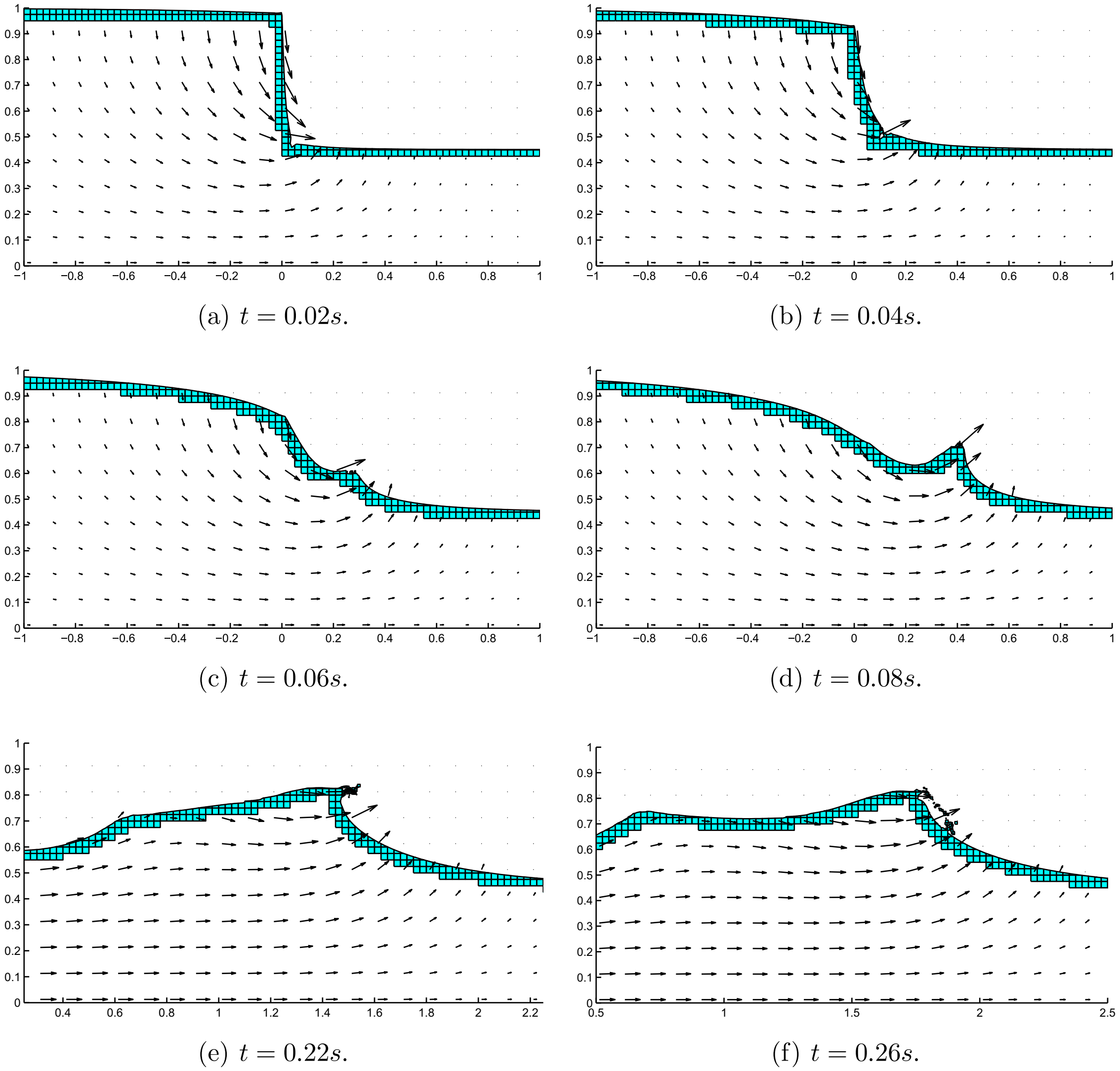
Fig. 13. Solitary wave form. The horizontal and vertical axes are $(x - x_{\text{crest}})/L_s$ and ζ/r_h, respectively. The solid line represents theoretical solution while '$\circ$', '+' and '$\cdot$' represent numerical results with $N = 5,10,20$, respectively.

Table 5

Convergence of solitary wave surface displacement.

	ϵ_ζ^5	ϵ_ζ^{10}	$\mathcal{O}^{10}$	ϵ_ζ^{20}	$\mathcal{O}^{20}$
$\bar{t} = 1.652$	6.20e−2	3.56e−2	0.80	1.60e−2	1.15
$\bar{t} = 2.478$	6.12e−2	3.95e−2	0.63	2.01e−2	0.98
$\bar{t} = 3.304$	6.75e−2	4.53e−2	0.57	2.08e−2	1.12
$\bar{t} = 4.130$	7.67e−2	4.55e−2	0.75	2.25e−2	1.02





(a) $t = 0.02s$.

(b) $t = 0.04s$.

(c) $t = 0.06s$.

(d) $t = 0.08s$.

(e) $t = 0.22s$.

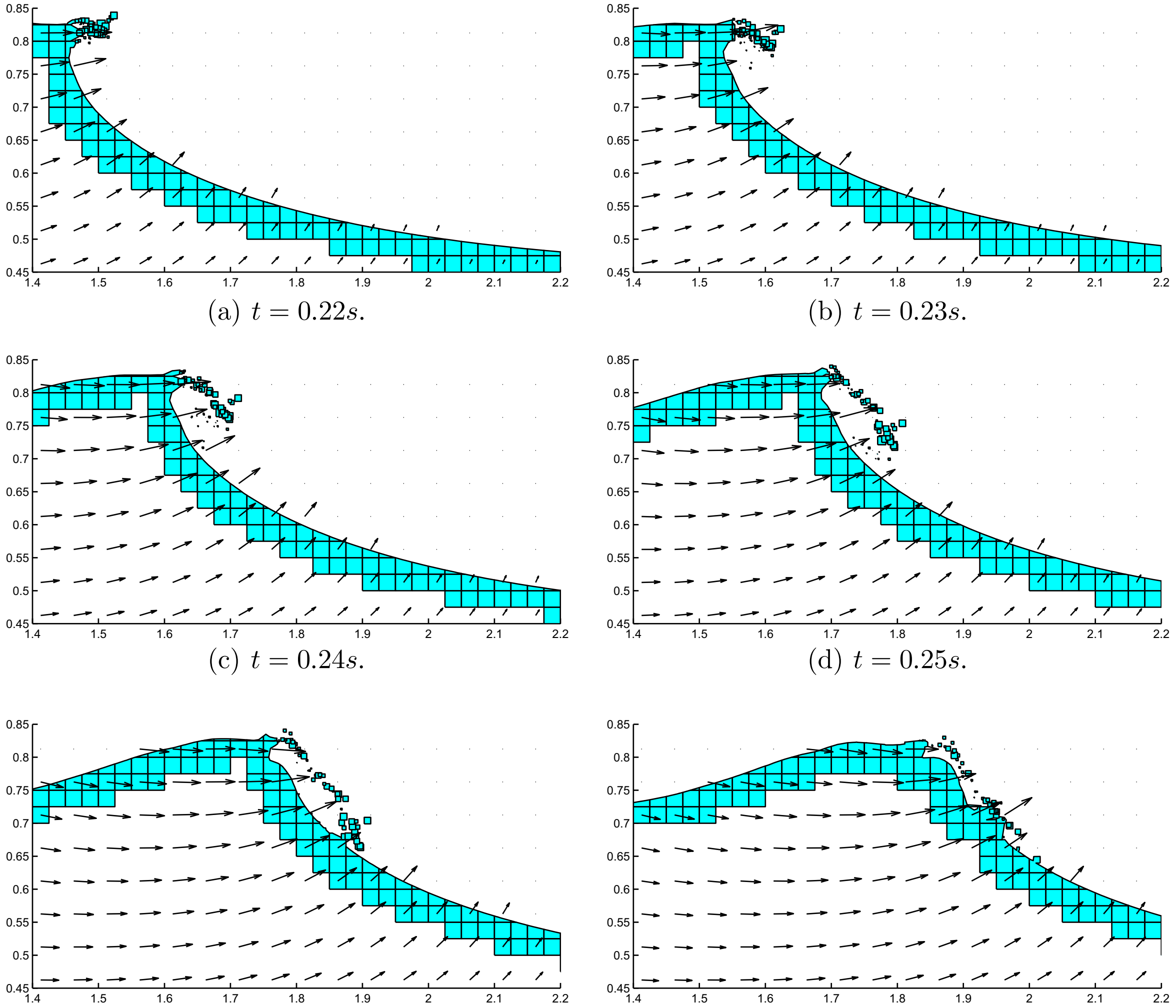
(f) $t = 0.26s$.

Fig. 15. Results of HyPAM for a dam-break test ($h_u = 0.10$ m, $\Delta x = \Delta y = h_u/40$). The axes are spatial coordinates normalized by h_u. Material polygons of the interface candidate cells are shown on actual grids and one velocity arrow is drawn for every 4-by-4 cells. Water particles are shown in small isolated squares.

causes an instantaneous adjustment of the pressure field. For example, the pressure along the line segment *AB* must change from hydrostatic to that of the air phase, thus pressure gradient become larger and larger from *A* to *B*. Consequently, water velocities become larger and larger from *A* to *B*, as shown in Fig. 15(a). In contrast, velocities at the gate-site below *B* are roughly the same because of constant pressure gradients across the gate. While the free surface far away from *AB* remains stationary, the surface at corner *A* falls vertically with gravitational acceleration, as shown in Fig. 15(a) and (b). Water particles initially at corner *B* undergoes very large accelerations and their velocities are always the largest in Fig. 15(a), (b) and (c), which produces a jet formation in Fig. 15(d). These results in the initial stage compare qualitatively to the potential-flow analysis in [80].

The jet formation in Fig. 15(d) evolves into a spilling breaker in Fig. 15(e) and (f). The detailed spilling process is shown again in Fig. 16 to illustrate the hybrid features of HyPAM. In Fig. 16(a) and (b), water polygons at the tip of the wave crest are separated from the water bulk. In Fig. 16(c) and (d), these water particles assume free fall with their initial velocities obtained by extrapolation from inside the water bulk. Later, they merge back into the water bulk in Fig. 16(e) and (f). Since the interaction of the particles is ignored, the algorithm essentially moves mass from one location to another until the interface topology and velocity field reach a balance on the connectedness of the free surface.

In VOF methods, volume fraction values in the 3-by-3 stencil centered at an interface cell are required to reconstruct its interface normal vector, usually approximated by taking gradients of the volume fraction function. If there are two or more



(a) $t = 0.22s$.

(b) $t = 0.23s$.

(c) $t = 0.24s$.

(d) $t = 0.25s$.

(e) $t = 0.26s$.

(f) $t = 0.27s$.

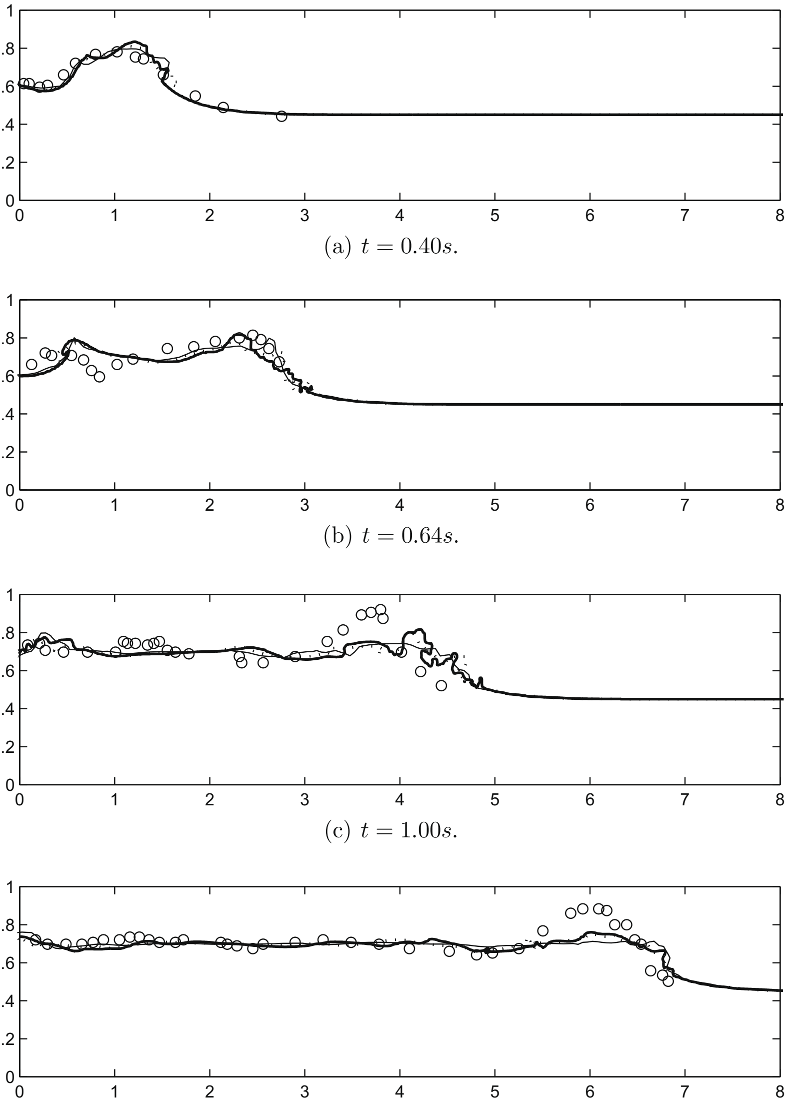
Fig. 16. Results of HyPAM for a spilling breaker. See Fig. 15 for caption.

distinct interface normals within this stencil, the reconstruction step generates spurious results with large errors. As such, the accuracy of VOF methods is limited by the size of the stencil and any droplet with its diameter smaller than $3h$ cannot be preserved. In fact, it is a common practice of VOF methods to complete remove a material from an interface cell if its volume fraction is smaller than a certain value, say, 0.05. Small particles tend to be consumed by this numerical artifact. Hence VOF methods are not suitable for hybrid continuum-particle formulations.

In contrast, the PAM method tracks interface details smaller than the grid size accurately, as shown in [96]. In Fig. 16, the particle shapes are well-preserved when a separation occurs. Essentially, the PAM method decouples the resolution of interface tracking from that of the flow field simulation in the HyPAM formulation. It is the resolution of the velocity field, not that of the PAM method, that limits the certainty of interface locations.

5.4.2. Quantitative validation: free surface locus

In Fig. 17, free surface results of COBRAS are compared to experimental data on three consecutively refined grids for the large-scale dam-break test. The results on the coarsest grids were also reported in [78]. At the initial stage, COBRAS agrees with experiments satisfactorily, as shown in Fig. 17(a). However, as the generated bore evolves with time, the discrepancy between COBRAS and experiments becomes larger and larger. In Fig. 17(c) and (d), COBRAS fails to predict the crests of the bore front. One reason for the low accuracy is that the VOF method used in COBRAS represents the interface by either a vertical or horizontal line. Another reason is that pure continuum methods do not have a mechanism to move particle mass from one place to another. Yet another more important reason is the velocity specification of 'newly-filled' interface cells. After updating the volume-fraction function, a pure air cell in previous time step might become an interface cell, thus velocity has to be specified at the cell faces for interface tracking. Chen [12] proposed the momentum-capturing method, which



(a) $t = 0.40s$.

(b) $t = 0.64s$.

(c) $t = 1.00s$.

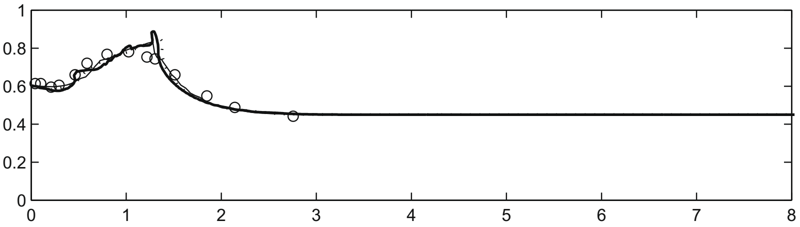
(d) $t = 1.44s$.

Fig. 17. Free surface results of COBRAS for a dam-break problem ($h_u = 0.36$ m). Both spatial coordinates are normalized by h_u. '○○': experimental results [80]; thin line: $\Delta x = \Delta y = h_u/20$; dotted line: $\Delta x = \Delta y = h_u/40$; thick line: $\Delta x = \Delta y = h_u/80$; all lines are volume-fraction contour plots of $f^W = 0.5$.

assumes the newly filled cell carries the momentum of its neighborhood and specifies the velocity of interface cells by constant extrapolation from inside the water phase. In COBRAS, this method is used. If the velocity variation near the free surface is negligible, this approximation is good enough. However, accuracy of this approximation is low for a free surface with large accelerations. Furthermore, temporal accumulation of errors results in non-convergence, as shown in Fig. 17(c) and (d), where refining the grids does not improve the accuracy.

In contrast, HyPAM yields much better results for exactly the same case in Fig. 18. The first reason is that the PAM method tracks the interface more accurately than VOF methods. The second reason is that the passive-response assumption leads to a better velocity field around the interface. The velocity specification of newly-filled cells are more physically consistent than the momentum-capturing method in that *both the momentum and the acceleration* are captured by the passive-response assumption. The third reason is that the hybrid formulation provides a way to move mass from one place to another, thus captures the physics of the spilling breaker that forms the wave crest, as shown in Fig. 16. These three advantages of HyPAM results in a much better prediction of the final wave crest, as shown in Fig. 18(d).

As noted in Section 1.2, the interface locus is uniquely determined by its initial position once the velocity field is given. Also determined are *whether* the interface will produce any particles and, if so, *how* the particles are detached. All physical factors influence these two aspects ('whether' and 'how') through the velocity field. For free-surface flows with high Weber numbers, surface tension is negligible comparing to other factors such as gravity, inertial and pressure gradient in



(a) $t = 0.40s$.

determining *whether* the interface will produce any particles. For example, the ‘jet’ topology shown in Fig. 15 is mainly due to the large pressure gradient. Nonetheless, when chunks of material are about to become particles, the local length-scale are such that surface tension must be important. In other words, surface tension plays an important role in determining *how* the particles are detached, e.g., the detailed interface topology and the shape of the particles. Realistic free-surface flows often feature a cascade of length-scales. For strongly fragmented free-surface flows, ignoring surface tension will introduce large errors since the dominant length-scale is small and most of the energy of the physical system is contained in this small length-scale; however, if the fragmentation is sporadic, the dominant length-scale is large and surface tension is trivial as far as the large length-scale is concerned. In this case, the ‘how’ issue is not as important as the ‘whether’ issue. From the above arguments, we emphasize that HyPAM is currently only applicable to mildly-fragmented free-surface flows.

Although the results in Fig. 18 are satisfactory, we point out the possible reasons of the disagreement between HyPAM and the experiments.

- In [80], other cases with different downstream water depth such as $h_d = 0.0h_u$ and $h_d = 0.1h_u$ are also studied. Although the free surfaces in the $h_d = 0.45h_u$ case are the least aerated, they still appear to be water/air mixtures with finite thickness. This introduces a certain degree of uncertainty since the free surface of the experimental results is defined as the surface bounding the water/air mixture.

- In the experiments, the gate is released not instantaneously, but in a finite period of time. As shown in [78], the releasing of the gate in a finite period has nontrivial effects on the free surface locus in subsequent times.
- The initial stage of dam-break flow is highly random in terms of free surface fluctuation and fragmentation. The experiments correspond to only one realization.

The randomness and finite releasing of the gate might explain the disagreement between HyPAM and the experiments in the initial stage. However, as the generated bore propagates, these effects tend to become unimportant. Indeed, the agreement between HyPAM and experiments are better in the final frame of Fig. 18 than all other frames.

6. Conclusion

Based on graph theory, we propose a new algorithm that decomposes a single phase into a continuum zone, a buffer zone and a particle zone. By exploiting topological information, this single-phase decomposition algorithm is able to deal with arbitrarily complicated transient free surface and it provides a robust foundation for many multi-phase formulations. As an example, HyPAM, a hybrid continuum-particle numerical model for two-dimensional incompressible free-surface flows with mild fragmentation, is formed by coupling this decomposition algorithm to the PAM method and a special pressure-incremental projection method that enforces the 'passive-response' assumption.

A droplet impact test is simulated to illustrate the main features of the new method while two dam break tests are studied to validate the physics of the numerical model. Mass and energy conservations show excellent convergence rate as the grid size is reduced. Numerical convergence of the flow field is shown by the droplet tests and that of interface tracking by the solitary wave propagation tests. By comparing our results to previous experimental and numerical work, we show that HyPAM is more accurate, versatile and efficient than VOF-based continuum methods for free-surface flows.

HyPAM is currently restricted to mildly-fragmented free-surface flows with large Reynolds numbers and Weber numbers; however, HyPAM has high potentials as a robust framework for multiphase flows. Including viscosity in the governing equation is trivial, due to the usage of the auxiliary velocity field $\tilde{\mathbf{u}}$. The extension of HyPAM to three dimension, is currently under progress. Although the projection method in HyPAM is only first-order accurate, an upgrade to second-order accuracy is fairly straightforward, see, e.g., [36,5,57,2,7]. The accurate implementation of surface tension is notoriously difficult for VOF methods, partly because the crude interface representation brings large errors into the evaluation of local curvatures. In contrast, under the framework of HyPAM, there are enough points in one interface cell to fit a surface for accurately evaluating the local curvature. Since surface tension results in a jump in the pressure field across the interface, one can either evaluate the surface tension force in the momentum equation, like the Immersed Boundary method [66,67], or implicitly incorporating the surface tension effect by solving a pressure Poisson equation with jumps in the solution, see [49] for an example. For a generalization to free-surface flows with strong fragmentation, the rigid-body free fall should be replaced with another particle method such as SPH or the LB method for the momentum coupling between the continuum zone and the particle zone.

Acknowledgments

We would like to acknowledge the supports from National Science Foundations through research grants to Cornell University. We also thank Prof. Stephen B. Pope, Prof. Stephen Vavasis and Prof. Edwin A. Cowen III for their valuable comments.

References

[1] A.V. Aho, J.E. Hopcroft, J.D. Ullman, Data Structures and Algorithms, Addison-Wesley, 1983.
[2] A.S. Almgren, J.B. Bell, P. Colella, L.H. Howell, M.L. Welcome, A conservative adaptive projection method for the variable density incompressible Navier–Stokes equations, J. Comput. Phys. 142 (1) (1998) 1–46.
[3] B. Ataie-Ashtiani, L. Farhadi, A stable moving-particle semi-implicit method for free surface flows, Fluid Dynamics Research 38 (2006) 241–256.
[4] S. Baase, Computer Algorithms: Introduction to Design and Analysis, Addison-Wesley, 1983.
[5] J.B. Bell, P. Colella, H.M. Glaz, A second-order projection method for the incompressible Navier–Stokes equations, J. Comput. Phys. 85 (2) (1989) 257–283.
[6] C.M. Bender, S.A. Orszag, Advanced Mathematical Methods for Scientists and Engineers, WKB Theory, McGraw-Hill Book Company, 1978 (Chapter 10).
[7] P.A. Berthelsen, O.M. Faltinsen, A local directional ghost cell approach for incomopressible viscous flow problems with irregular boundaries, J. Comput. Phys. 227 (2008) 4354–4397.
[8] M. Brocchini, D.H. Peregrine, The dynamics of strong turbulence at free surface. Part 1. Description, J. Fluid Mech. 449 (2001) 225–254.
[9] D.L. Brown, R. Cortez, M.L. Minion, Accurate projection methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 168 (2) (2001) 464–499.
[10] R.K.C. Chan, R.L. Street, A computer study of finite amplitude water waves, J. Comput. Phys. 6 (1970) 68–94.
[11] H. Chanson, Air Bubble Entrainment in Free-surface Turbulent Shear Flows, Academic Press, London, UK, 1996, ISBN 0121681106.
[12] S. Chen, D.B. Johnson, P.E. Raad, Velocity boundary conditions for the simulation of free surface fluid flow, J. Comput. Phys. 116 (2) (1995) 262–276.
[13] A.J. Chorin, Numerical solution of the Navier–Stokes equations, Math. Comput. 22 (104) (1968) 745–762.
[14] A.J. Chorin, On the convergence of discrete approximations to the Navier–Stokes equations, Math. Comput. 23 (106) (1969) 341–353.
[15] A. Colagrossi, M. Landrini, Numerical simulation of interfacial flows by smoothed particle hydrodynamics, J. Comput. Phys. 191 (2003) 448–475.
[16] V. Cristini, J. Blawzdziewicz, M. Loewenberg, An adaptive mesh algorithm for evolving surfaces: Simulations of drop breakup and coalescence, J. Comput. Phys. 168 (2) (2001) 445–463.
[17] J. Cui, G.W. He, D. Qi, A constrained particle dynamics for continuum-particle hybrid method in micro- and nano-fluidics, Acta Mech. Sin. 22 (2006) 503–508.

[18] M. Dai, D.P. Schmidt, Adaptive tetrahedral meshing in free-surface flow, J. Comput. Phys. 208 (2005) 228–252.

[19] R.G. Dean, R.A. Dalrymple, Water Wave Mechanics for Engineers and Scientists, World Scientific, Singapore, 1991.

[20] R. Delgado-Buscalioni, P.V. Coveney, Continuum-particle hybrid coupling for mass, momentum, and energy transfers in unsteady fluid flow, Phys. Rev. E 67 (4) (2003) 046704.

[21] R. Diestel, Graph Theory, Springer-Verlag, Heidelberg, New York, 2005.

[22] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface tracking, J. Comput. Phys. 183 (2002) 83–116.

[23] D. Enright, S. Marschner, R. Fedkiw, Animation and rendering of complex water surfaces, ACM Trans. Graph. 21 (3) (2002) 736–744.

[24] K.A. Flack, J.R. Sylor, G.B. Smith, Near-surface turbulence for evaporative convection at an air/water interface, Phys. Fluids 13 (11) (2001) 3338.

[25] N. Foster, R. Fedkiw. Practical animation of liquids, in: SIGGRAPH 2001 Annual Conference, 2001, pp. 15–22.

[26] J. Fukai, Z. Zhao, D. Poulikakos, C.M. Megaridis, O. Miyatake. Modeling of the deformation of a liquid droplet impinging upon a flat surface, J. Comput. Phys. 5(11), Phys. Fluids A – Fluid Dynam., 1993.

[27] I. Ginzburg, K. Steiner, Lattice Boltzmann model for free-surface flow and its application to filling process in casting, J. Comput. Phys. 185 (2003) 61–99.

[28] D. Gonzalez, E. Cueto, F. Chinesta, M. Doblare, A natural element updated lagrangian strategy for free-surface fluid dynamics, J. Comput. Phys. 223 (2007) 127–150.

[29] M. Gorokhovski, M. Herrmann, Modeling primary atomization, Annu. Rev. Fluid Mech. 40 (2008) 343–366.

[30] R. Grimshaw, The solitary wave in water of variable depth. part 2, J. Fluid Mech. 46 (3) (1971) 611–622.

[31] J.L. Gross, J. Yellen, Graph Theory and its Applications, Chapman & Hall/CRC, Boca Raton, London, New York, 2006.

[32] C.W. Hirt, B.D. Nichols, Volume of fluid (vof) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1) (1981) 201–225.

[33] H.H. Hu, N.A. Patankar, M.Y. Zhu, Direct numerical simulations of fluid–solid systems using the arbitrary lagrangian–eulerian technique, J. Comput. Phys. 169 (2) (2001) 427–462.

[34] J.C.R. Hunt, J.M.R. Graham, Free-stream turbulence near plane boundaries, J. Fluid Mech. 84 (2) (1978) 209–235.

[35] T. Inamuro, T. Ogata, S. Tajima, N. Konishi, A lattice Boltzmann method for incompressible two-phase flows with large density differences, J. Comput. Phys. 198 (2004) 628–644.

[36] J. Kim, P. Moin, Application of a fractional-step method to incompressible Navier–Stokes equations, J. Comput. Phys. 59 (2) (1985) 308–323.

[37] S. Koshizuka, A. Nobe, Y. Oka, Moving-particle semi-implicit method for fragmentation of incompressible fluids, Int. J. Numer. Meth. Fluids 26 (1998) 751–769.

[38] S. Koshizuka, Y. Oka, Moving-particle semi-implicit method for fragmentation of incompressible fluids, Nucl. Sci. Eng. 123 (1996) 421–434.

[39] D.B. Kothe, R.C. Mjolsness, Ripple: A new model for incompressible flows with free surfaces, AIAA J. 30 (11) (1992) 2694–2700.

[40] P. Koumoutsakos, Multiscale flow simulations using particles, Annu. Rev. Fluid Mech. 37 (2005) 457–487.

[41] C.M. Lemos, Wave Breaking: A Numerical Study, Lecture Notes in Engineering, vol. 71, Springer-Verlag, Berlin, New York, 1992.

[42] R.J. Leveque, Z. Li, Immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (4) (1994) 1019–1044.

[43] R.J. Leveque, Z. Li, Immersed interface methods for stokes flow with elastic boundaries or surface tension, SIAM J. Sci. Comput. 18 (3) (1997) 709–735.

[44] Z. Li, M.-C. Lai, The immersed interface method for the navier-stokes equations with singular forces, J. Comput. Phys. 171 (2) (2001) 822–842.

[45] P. Lin, P.L.-F. Liu, Numerical study of breaking waves in the surf zone, J. Fluid Mech. 359 (1998) 239–264.

[46] D. Liu, P. Lin, A numerical study of three-dimensional liquid sloshing in tanks, J. Comput. Phys. 227 (2008) 3921–3939, doi:10.1016/j.jcp.2007.12.006.

[47] J. Liu, S. Chen, X. Nie, M.O. Robbins, A continuum-atomistic simulation of heat transfer in micro- and nano-flows, J. Comput. Phys. 227 (2007) 279–291.

[48] J. Liu, S. Koshizuka, Y. Oka, A hybrid particle-mesh method for viscous, incompressible, multiphase flows, J. Comput. Phys. 202 (2005) 65–93.

[49] X.-D. Liu, R.P. Fedkiw, M. Kang, A boundary condition capturing method for poisson's equation on irregular domains, J. Comput. Phys. 160 (2000) 151–178.

[50] D.C. Lo, D.L. Young, Arbitrary lagrangian-eulerian finite element analysis of free surface flow using a velocity-vorticity formulation, J. Comput. Phys. 195 (2004) 175–201.

[51] M.S. Longuet-Higgins, Mass transport in water waves, Phil. Trans. R. Soc. Lond. A 245 (903) (1953) 535–581.

[52] M.S. Longuet-Higgins, Mass transport in the boundary layer at a free oscillating surface, J. Fluid Mech. 8 (1960) 293–306.

[53] M.S. Longuet-Higgins, Capillary rollers and bores, J. Fluid Mech. 240 (1992) 659–679.

[54] F. Losasso, J.O. Talton, N. Kwatra, R. Fedkiw, Two-way coupled sph and particle level set fluid simulation, IEEE Trans. Vis. Comput. Graph. (2008), doi:10.1109/TVCG.2008.37.

[55] C.C. Mei, M. Stiassnie, D.K.-P. Yue, Theory and Applications of Ocean Surface Waves, Part 1: Linear Aspects, World Scientific, Singapore, 2005.

[56] W.K. Melville, The role of surface-wave breaking in air–sea interaction, Annu. Rev. Fluid Mech. 28 (1996) 279–321.

[57] M.L. Minion, A projection method for locally refined grids, J. Comput. Phys. 127 (1) (1996) 158–178.

[58] J.J. Monaghan, Smoothed particle hydrodynamics, Annu. Rev. Astron. Astrophys. 30 (1992) 543–574.

[59] J.J. Monaghan, Simulating free surface flows with sph, J. Comput. Phys. 110 (1994) 399–406.

[60] S.T. O'Connell, P.A. Thompson, Molecular dynamics-continuum hybrid computations: a tool for studying complex fluid flows, Phys. Rev. E 52 (6) (1995) 5792–5795.

[61] G. Oger, M. Doring, B. Alessandrini, P. Ferrant, An improved sph method: towards higher order convergence, J. Comput. Phys. 225 (2007) 1472–1492.

[62] J. O'Rourke, Computational Geometry in C, second ed., Cambridge University Press, New York, 1998.

[63] R.L. Panton, Incompressible Flow, John Wiley & Sons INC., New York, 1996.

[64] E.I. Parau, J.-M. Vanden-Broeck, M.J. Cooker, Nonlinear three-dimensional interfacial flows with a free surface, J. Fluid Mech. 591 (2007) 481–494.

[65] B. Perot, R. Nallapati, A moving unstructured staggered mesh method for the simulation of incompressible free-surface flows, J. Comput. Phys. 184 (2003) 192–214.

[66] C.S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (1972) 252–271.

[67] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.

[68] T. Sarpkaya, Vorticity, free surface, and surfactants, Annu. Rev. Fluid Mech. 28 (1996) 83–128.

[69] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow calculations, Ann. Rev. Fluid Mech. 31 (1999) 567–603.

[70] S. Schaffer, A semi-coarsening multigrid method for elliptic partial differential equations with highly discontinuous and anisotropic coefficients, SIAM J. Sci. Comput. 20 (1) (1998) 228–242.

[71] T.E. Schwartzentruber, L.C. Scalabrin, I.D. Boyd, A modular particle-continuum numerical method for hypersonic non-equilibrium gas flows, J. Comput. Phys. 225 (2007) 1159–1174.

[72] J.C. Scott, The preparation of water for surface-clean fluid mechanics, J. Fluid Mech. 69 (2) (1975) 339–351.

[73] A. Selle, N. Rasmussen, R. Fedkiw, A vortex particle method for smoke water and explosions, ACM Trans. Graph. 24 (3) (2005) 910–914.

[74] J.A. Sethian, P. Smereka, Level-set methods for fluid interfaces, Ann. Rev. Fluid Mech. 35 (2003) 341–372.

[75] L. Shen, G.S. Triantafyllou, D.K.P. Yue, Turbulent diffusion near a free surface, J. Fluid Mech. 407 (2000) 145–166.

[76] L. Shen, D.K.P. Yue, G.S. Triantafyllou, Effect of surfactants on free-surface turbulent flows, J. Fluid Mech. 506 (2004) 79–115.

[77] L. Shen, X. Zhang, D.K.P. Yue, G.S. Triantafyllou, the surface layer for free-surface turbulent flows, J. Fluid Mech. 386 (1999) 167–212.

[78] T. Shigematsu, P.L.-F. Liu, K. Oda, Modelisation numerique des etapes initiales de l'onde de rupture de barrage; numerical modeling of the initial stages of dam-break waves, J. Hydraul. Res. 42 (2) (2004) 183–195.

[79] J.G. Siek, L.-Q. Lee, A. Lumsdaine. The Boost Graph Library, C++ In-depth Series, Addison-Wesley, 2002, ISBN 0-201-72914-8.

[80] P.K. Stansby, A. Chegini, T.C.D. Barnes, Initial stages of dam-break flow, J. Fluid Mech. 374 (1998) 407–424.

[81] K.-I. Sugioka, S. Komori, Drag and lift forces acting on a spherical water droplet in homogeneous linear shear air flow, J. Fluid Mech. 570 (2007) 155–175.

[82] R.E. Tarjan, Depth first search and linear graph algorithms, SIAM J. Comput. 1 (2) (1972) 146–160.

[83] S.A. Thorpe, Dynamical processes of transfer at the sea surface, Prog. Oceanogr. 35 (4) (1995) 315–352.

[84] K. Thulasiraman, M.N.S. Swamy, Graphs: Theory and Algorithms, John Wiley & Sons, 1992.

[85] B.R. Vatti, A generic solution to polygon clipping, Commun. ACM 35 (7) (1992) 56–63.

[86] E. Villermaux, Fragmentation, Annu. Rev. Fluid Mech. 39 (2007) 419–446, doi:10.1146/annurev.fluid.39.050905.110214.

[87] Y. Watanabe, A. Saruwatari, D.M. Ingram, Free-surface flows under impacting droplets, J. Comput. Phys. 227 (2008) 2344–2365.

[88] S.W.J. Welch, Local simulation of two-phase flows including interface tracking with mass-transfer, J. Comput. Phys. 121 (1) (1995) 142–154.

[89] C.E. Willert, M. Gharib, The interaction of spatially modulated vortex pairs with free surfaces, J. Fluid Mech. 345 (1997) 227–250.

[90] S.A. Williams, J.B. Bell, A.L. Garcia, Algorithm refinement for fluctuating hydrodynamics, Multiscale Model. Simul. 6 (2008) 1256–1280.

[91] X. Xie, L.C. Musson, M. Pasquali, An isochoric domain deformation method for computing steady free surface flows with conserved volumes, J. Comput. Phys. 226 (2007) 398–413.

[92] S. Xu, Z.J. Wang, Systematic derivation of jump conditions for the immersed interface method in three-dimensional flow simulation, SIAM J. Sci. Comput. 27 (6) (2006) 1948–1980.

[93] T. Yabe, F. Xiao, T. Utsumi, The constrained interpolation profile method for multiphase analysis, J. Comput. Phys. 169 (2001) 556–593.

[94] B. Yang, A. Prosperetti, A second-order boundary-fitted projection method for free-surface flow computations, J. Comput. Phys. 213 (2006) 574–590.

[95] C. Zhang, L. Shen, D.K.P. Yue, The mechanism of vortex connection at a free surface, J. Fluid Mech. 384 (1999) 207–241.

[96] Q. Zhang, P.L.-F. Liu, A new interface tracking method: The polygonal area mapping method, J. Comput. Phys. 227 (8) (2008) 4063–4088, doi:10.1016/j.jcp.2007.12.014.

[97] Q. Zhang, P.L.-F. Liu, A numerical study of swash flows generated by bores, Coastal Eng., 2009, in press, doi:10.1016/j.coastaleng.2008.04.010.

[98] A.Z. Zinchenko, M.A. Rother, R.H. Davis, A novel boundary-integral algorithm for viscous interaction of deformable drops, Phys. Fluids A – Fluid Dynam. 9 (6) (1997) 1493–1511.