An Extended Cut-Cell Method for Sub-Grid Liquids Tracking with Surface Tension

YI-LU CHEN, ETH Zurich
JONATHAN MEIER, ETH Zurich
BARBARA SOLENTHALER, ETH Zurich
VINICIUS C. AZEVEDO, ETH Zurich

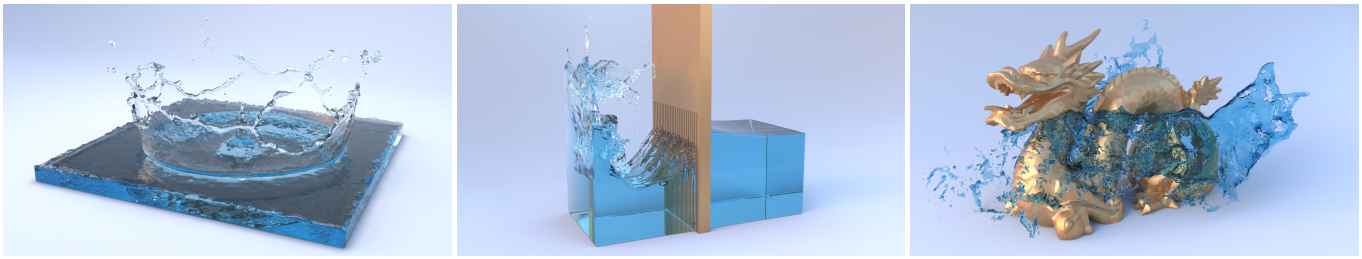


Fig. 1. Our cut-cells solver allows detailed fluid simulation in coarse regular grids, handling surface tension, complex obstacles, and thin liquid sheets.

Simulating liquid phenomena utilizing Eulerian frameworks is challenging, since highly energetic flows often induce severe topological changes, creating thin and complex liquid surfaces. Thus, capturing structures that are small relative to the grid size become intractable, since continually increasing the resolution will scale sub-optimally due to the pressure projection step. Previous methods successfully relied on using higher resolution grids for tracking the liquid surface implicitly; however this technique comes with drawbacks. The mismatch of pressure samples and surface degrees of freedom will cause artifacts such as hanging blobs and permanent kinks at the liquid-air interface. In this paper, we propose an extended cut-cell method for handling liquid structures that are smaller than a grid cell. At the core of our method is a novel iso-surface Poisson Solver, which converges with second-order accuracy for pressure values while maintaining attractive discretization properties such as symmetric positive definiteness. Additionally, we extend the iso-surface assumption to be also compatible with surface tension forces. Our results show that the proposed method provides a novel framework for handling arbitrarily small splashes that can also correctly interact with objects embodied by complex geometries.

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

Additional Key Words and Phrases: physically-based simulations, fluid animation, natural phenomena

Authors' addresses: Yi-Lu Chen, ETH Zurich, chyi@student.ethz.ch; Jonathan Meier, ETH Zurich, jonathan.meier@outlook.com; Barbara Solenthaler, ETH Zurich, solenthaler@inf.ethz.ch; Vinicius C. Azevedo, ETH Zurich, vinicius.azevedo@inf.ethz.ch.

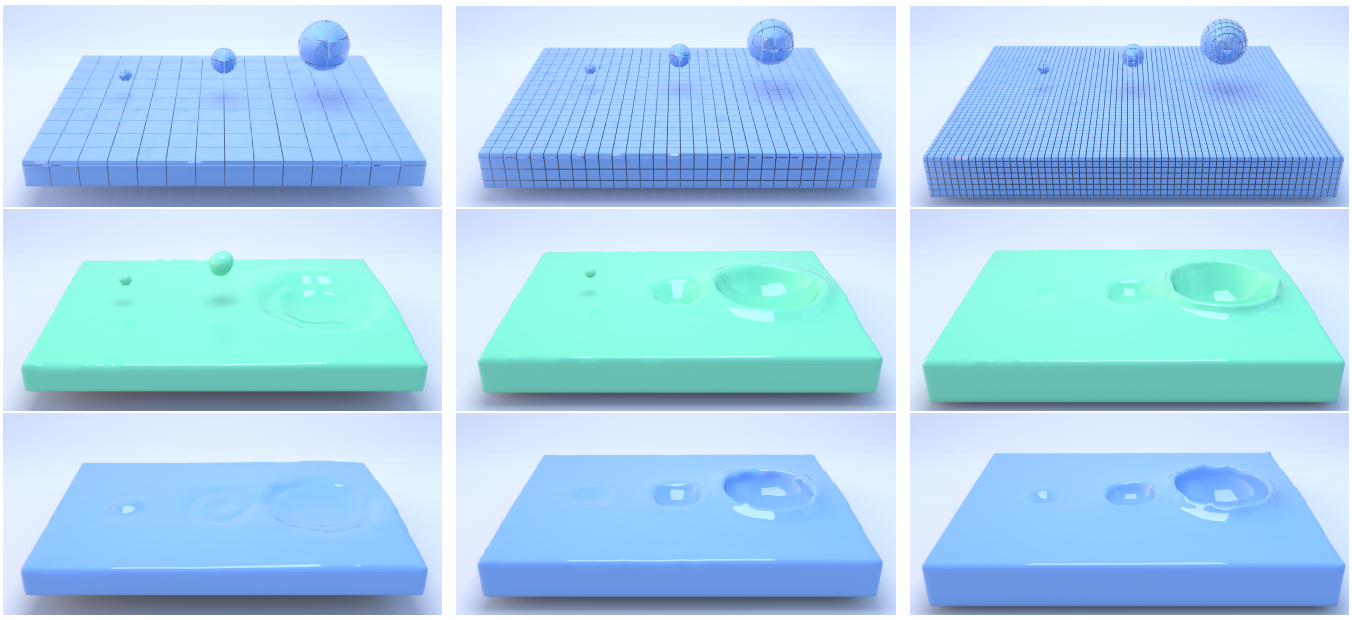
0730-0301/2020/12-ART169 $15.00
https://doi.org/10.1145/3414685.3417859

ACM Reference Format:
Yi-Lu Chen, Jonathan Meier, Barbara Solenthaler, and Vinicius C. Azevedo. 2020. An Extended Cut-Cell Method for Sub-Grid Liquids Tracking with Surface Tension. *ACM Trans. Graph.* 39, 6, Article 169 (December 2020), 13 pages. https://doi.org/10.1145/3414685.3417859

1 INTRODUCTION

Visually compelling liquid phenomena involve thin and complex water splashes. Fluids split and merge, undergoing through severe topological changes due to large bodies of water breaking down into smaller ones, represented by surfaces in which elements' scale varies substantially. The dynamic behavior of liquids creates small intricately connected elements that are challenging to model with standard regular grid discretizations. Continuously increasing the resolution until small-scale details are tractable is often expensive and impractical for most animation scenarios. Additionally, highly refined grids may require prohibitively small time steps to avoid instabilities and to correctly preserve details.

Volumetric grids scale poorly when tracking two-dimensional surfaces and it is common practice to use higher grid resolutions for tracking the liquid's implicit representation [Bargteil et al. 2006; Goktekin et al. 2004] without refining the underlying grid used for the simulation. The problem with such methods is that sub-grid information is invisible to the fluid solver, and many artifacts, such as floating blobs, kinks, and tiny air bubbles can permanently remain on the liquid surface. Previous works have attempted to solve this either by surface smoothing [Kim et al. 2009; Wojtan and Turk 2008], energy minimization [Bojsen-Hansen and Wojtan 2013; Goldade et al. 2016], or by explicitly coupling the underlying regular representation to the liquid surface through meshing [Brochu et al. 2010; English et al. 2013]. However, these approaches fail to capture the topologically complex liquid domain in the presence of narrow air gaps or thin liquid sheets, or require expensive meshing or coupling algorithms to succeed.



(a) $16 \times 8 \times 12$ (b) $32 \times 16 \times 24$ (c) $64 \times 32 \times 48$

Fig. 2. A comparison of falling drops with initial state (top) in a coarse fluid simulation using the Ghost Fluid Method (middle), and our method (bottom) across different resolutions. When the surface tracker is much finer than the pressure grid, details become invisible to the simulation, causing small blobs to hang in mid air with the Ghost Fluid Method. Our method captures such small details in all three resolutions, correctly modelling splashing liquid blobs.

Thus, we propose an Eulerian fluid solver which captures subgrid features such as thin liquid splashes, narrow air spaces and droplets on regular grids (Figure 2), while still largely conserving the volumes of the embedded liquids. Since our approach is an extension of the cut-cells method [Azevedo et al. 2016], it can also naturally handle liquids interacting with complex obstacles (Figures 3 and 6). Our method is enabled by a novel Laplacian solver that handles non grid-aligned Dirichlet boundary conditions. It extends the traditional Ghost Fluid Method pressure Dirichlet condition at the liquid-air interface through a novel *iso-surface* assumption: up to a pre-specified distance to the boundary, all pressures have the same value. Thus, only one value will be used to model pressures in different spatial locations (Figure 7) per cut-cell, reducing the computational burden of modelling potentially highly-detailed liquid surfaces to a single variable. We show that our method tracks thin liquid sheets (Figure 11) and narrow gaps (Figure 3), while retaining the efficiency of Cartesian grids, yielding a second-order accurate solver and a symmetric positive definite formulation.

The presence of interfacial surfaces forces, such as surface tension, creates another challenge. The original iso-surface assumption is no longer valid and we augment our method with an independent pressure scalar field at the liquid-air interface, decoupling these pressures from the ones inside the liquid. Our method is computationally attractive, since the decoupled pressure system relaxes the usual severe time-step limitations when evaluating liquid surface tension forces. In summary, the contributions of this paper are:

- A novel iso-surface Poisson solver capable of modelling Dirichlet boundary conditions for complex topologies embedded in regular grids. The resulting linear system is symmetric positive definite, while exhibiting second-order accurate pressure and first-order accurate gradient values (Section 3);
- An extended cut-cells surface tension solver (Section 4);

2 RELATED WORK

Since the pioneering work of Foster and Metaxas [1996], fluids in Computer Graphics have thoroughly evolved to a mature state and are widely employed in production pipelines for digital content creation [Frost et al. 2017; Stomakhin and Selle 2017]. Specific to liquids, interface tracking can be subdivided into implicit [Enright et al. 2002; Foster and Fedkiw 2001] and explicit representations [Wojtan et al. 2009, 2010]. Implicit liquid tracking uses a level-set function to represent interfaces in regular grids, while explicit representations track vertex positions of triangle meshes. Implicit representations can easily handle topological changes such as merging and splitting and are therefore more commonly adopted. The implicit liquid representation can be constructed from particles [Ando et al. 2012; Jiang et al. 2015; Zhu and Bridson 2005] or tracked fully in an Eulerian setting [Heo and Ko 2010]. Although in this work we adopt an implicit representation, explicit surface tracking can also be handled by our cut-cells algorithm.

Adopting *higher grid resolutions for surface tracking* is a useful technique to overcome detail loss in implicit representations, and was firstly employed to model viscoelastic fluids [Goktekin et al. 2004]. Bargteil et al. [2006] combined semi-Lagrangian contouring with adaptive level-sets near surfaces, and impressive liquid simulations were obtained by super-sampling liquid geometries

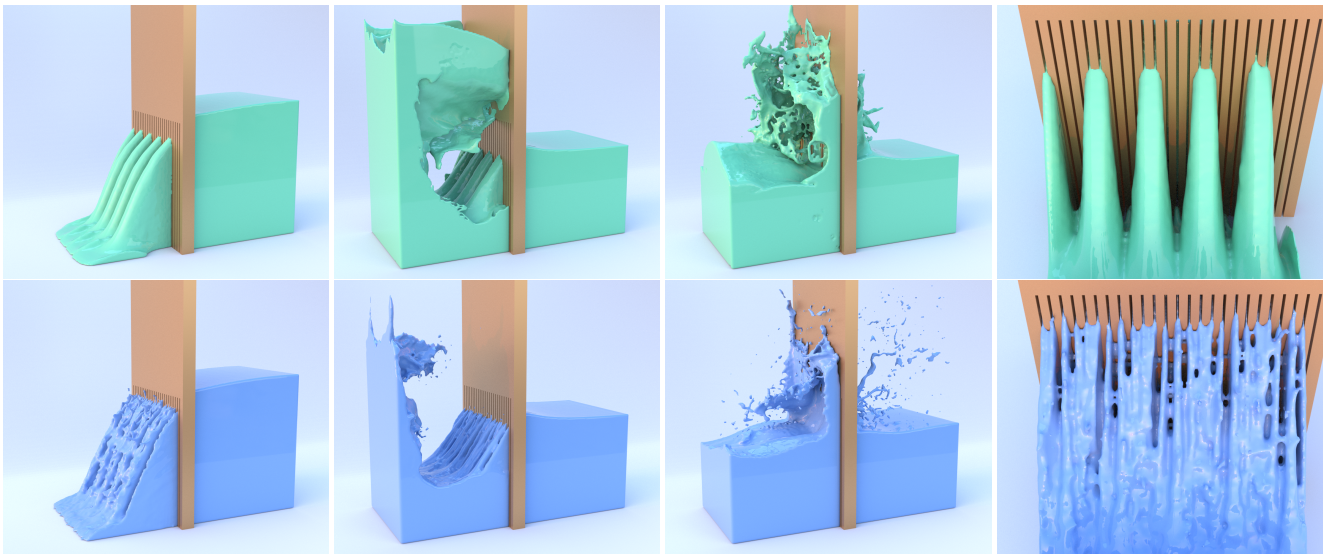


Fig. 3. Liquid flowing through narrow slits using a grid resolution of $64 \times 64 \times 32$. While our method reliably detects all gaps and allows liquid to flow through, the GFM can only detect slits if they are aligned with the grid cells, leading to fewer and wider gaps. The wider gaps modelled by the GFM will cause the fluid to incorrectly flow faster through the slits.

with volume-preserving liquid-biased filtering [Kim et al. 2009]. To remove persistent artifacts created by the resolution mismatch while still keeping interesting mesh details, Bojsen-Hansen and Wojtan [2013] proposed an energy function embedded in the lower resolution simulation grid, which can be used to either eliminate the artifacts via energy minimization or use these errors for sub-grid simulations. Goldade et al. [2016] further simplified this idea via Gaussian pyramid filtering and improved the conditions of liquid tracking in narrow-band settings. As an alternative to directly solving a pressure field in lower dimensions, Ando et al [2015] proposed solving a dimension-reduced linear system, which after one linear transformation simultaneously approximates the pressure and enforces the pressures at the liquid–air interface in the higher dimensions. However, this simplification does not exactly enforce incompressibility at the liquid-air interface, contrary to our proposed approach. Higher resolution surface tracking is also employed not only for regular grid simulations. Sifakis et al. [2007] developed a method that embedded a high-resolution point-sampled surface in a coarse finite element mesh and Wojtan and Turk [2008] modelled viscoelastic behavior by combining a high resolution surface tracker with a lower resolution tetrahedral finite element simulator.

Fractional boundaries and cut-cells are employed to embed or explicitly track sub-grid details in a regular grid solver. Batty et al. [2007] reformulated the classic pressure projection step as an energy minimization problem, deriving the resulting boundary conditions as fractional boundaries. Further work [Ng et al. 2009] identified this method as not convergent, and an improved boundary handling formulation was proposed to model arbitrary solid geometry while maintaining a second-order symmetric positive-definite (SPD) discretization. Stronger two-way coupling between liquids and deformables [Zarifi and Batty 2017] and viscous liquids and solids [Takahashi and Lin 2019] extended previous fractional boundaries approaches. Cut-cells [Azevedo et al. 2016; Edwards and Bridson 2014] detach distinct regions of the flow separated by boundaries in a topologically robust way, capturing arbitrarily thin boundaries and narrow gaps. Edwards et al. [2014] coupled cut-cells with a p-adaptive Discontinuous Galerkin method for detailed water capturing. However, this method is computationally expensive and complex to implement. Simpler cut-cell methods were used to model interaction of solids and fluids [Azevedo et al. 2016], while also improving velocity interpolation close to boundaries through Spherical Barycentric Coordinates [Langer et al. 2006] interpolants. Robustly generating cut-cells can be a burdensome task, and Tao et al. [2019] developed an open-source framework for cut-cells creation. Lastly, topologically accurate finite element methods were also employed to model deformable bodies [Nesme et al. 2009] and musco-skeletal [Teran et al. 2005] simulations.

Adaptive refining and meshing. Poor volumetric scaling of 3D grids for tracking 2D liquid surfaces motivated researchers to develop adaptive simulation algorithms. Octrees can provide efficient spatial refinement, and were thus used to represent liquids [Losasso et al. 2004], smoke [Shi and Yu 2004], surface tension [Hong and Kim 2005], bubbles [Kim et al. 2007], viscous liquids [Goldade et al. 2019] and were tailored for efficiency [Aanjaneya et al. 2017; Setaluri et al. 2014]. While these methods concentrate variables near the free surfaces, they break the grid's regularity, and the discretization accuracy has to be sacrificed for maintaining positive-definite systems. Further work improved the accuracy of gradients on T-junctions [Batty 2017] and obtained improved SPD discretizations [Goldade et al. 2019]. Adaptive meshes are not restricted to regular grids. Several works [Ando et al. 2013; Batty and Houston 2011;

Batty et al. 2010; Chentanez et al. 2007; Klingner et al. 2006] use adaptivity in tetrahedral meshes to increase accuracy and preserve details in regions of interest. Virtual Node Methods [Bedrossian et al. 2010; Hellrung et al. 2012] are an alternative solution for Poisson equations in irregular domains with second order accuracy, however these methods do no inherently support complex sub-grid phenomena such as tiny droplets or thin liquid splashes. Chimera [English et al. 2013] and Arbitrary Lagrangian-Eulerian grids [Lee et al. 2019] are also employed for improving liquid tracking in regions of interest. Using adaptive mesh elements is also common in simulation of diverse physical phenomena such as cloth [Narain et al. 2012], paper folding and tearing [Narain et al. 2013], fractures [Molino et al. 2004], elastoplastic materials [Wicke et al. 2010], and multiphase fluids [Misztal et al. 2012].

Surface Tension. For Eulerian simulations, Kang et al. [2000] provided a popular method to calculate the necessary boundary conditions by incorporating pressure jumps into the Ghost Fluid method, while Müller et al. [2003] discussed surface tension with Smooth Hydrodynamic Particles. An inherent challenge to adding surface tension, however, is the increased restriction in time step size. Volume-preserving mean curvature flow in calculating the boundary conditions was used to effectively reduce the timestep restriction from $O(\Delta x^{3/2})$ back to $O(\Delta x)$ [Sussman and Ohta 2009]. Several methods also explored fully or partially decoupling surface tension from fluid pressure forces. Cohen and Molemaker [2004] utilized several surface tension-driven advection sub-steps in one simulation step. Thürey et al. [2010] decoupled surface tension by performing a mesh-based wave simulation at every frame to simulate capillary waves and to increase stability. Since surface tension phenomena are most prominent in small scale liquids, other specialized techniques have been proposed to simulate specific phenomena. Wang et al. [2005] explored the simulation of contact angles on the solid-liquid interface by the use of a virtual level set which penetrates the solid surface. Methods for froth [Cleary et al. 2007; Kim et al. 2010], foam [Kim et al. 2007], and bubbles [Da et al. 2015] also make use of surface tension forces to increase simulation fidelity. More recently, [Batty et al. 2012] used a reduced order model to simulate the dynamics of thin sheets of viscous incompressible liquid with surface tension incorporated by nonlinear forces by minimizing the discrete surface areas. The surface-only liquid simulation by Da et al. [2016] deals with such settings, describes the liquid with only the triangle mesh as a Lagrangian representation of the liquid surface, and enforces the incompressibility constraint accordingly.

2.1 Cut-Cells For Solid Embedded Geometries

We directly extend the cut-cells method [Azevedo et al. 2016], which supports accurate handling of sub-grid information in potentially coarse resolutions. Hence, we will detail its analysis of solid boundary conditions, which is relevant for this work. To generate cut-cells, regular grid cells are intersected with obstacles embedded in the simulation, and a single regular cell can be potentially subdivided into many cut-cells. Each cut-cell will store its own pressure and velocity, and arbitrarily small gaps and thin meshes can be represented in a topologically correct way.

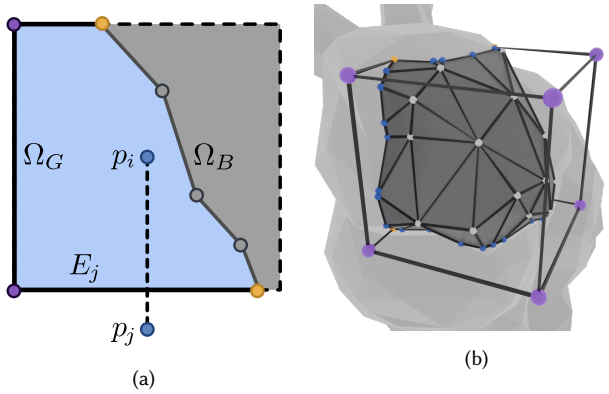


Fig. 4. (a): A diagram of a 2-D cut cell with different components highlighted. Purple: grid vertices, yellow: edge vertices, gray: geometry vertices. The cell is connected to the cell below via edge E_j, thus a pressure gradient is calculated with p_i and p_j. (b): An illustration of a 3D cut cell, with the additional face vertices in blue and the geometry faces highlighted.

2.1.1 Cut-Cells Structures. We further introduce the categorization of various cut-cells sub-components - vertices, edges and faces - which will be referenced by our extended cut-cell method. Cut-cell vertices are categorized by their spatial location as

- A *grid vertex* is a vertex that coincides with a vertex of the underlying regular grid;
- An *edge vertex* is a vertex that simultaneously lies on an edge of the regular grid and on the embedded geometry;
- A *face vertex* is a vertex that simultaneously lies on a face of the regular grid and on the embedded geometry [1];
- A *geometry vertex* refers to a vertex that lies exclusively on the embedded geometry.

Cut-cell *boundary structures* - represented by *edges in 2-D* and *faces in 3-D* - are also categorized by their spatial locations as

- A cut-cell *grid edge (2-D) or grid face (3-D)* refers to an edge/face that coincides with the corresponding element on the underlying regular grid. These structures are formed when a regular cell is intersected with the embedded geometry. Two cut-cells that share the same grid edge (2-D) or grid face (3-D) have connected pressure samples. We further refer the collection of those elements contained in a particular cell by Ω_G.
- A cut-cell *geometry edge (2-D) or geometry face (3-D)* refers to an edge/face which exclusively lies on the embedded geometry. These structures are usually not aligned with the regular grid and we further refer to the collection of those elements contained in a particular cell as Ω_B [2].

2.1.2 Boundary Conditions for Embedded Solid Geometries. The cut-cell method employs a finite-volume discretization of the pressure

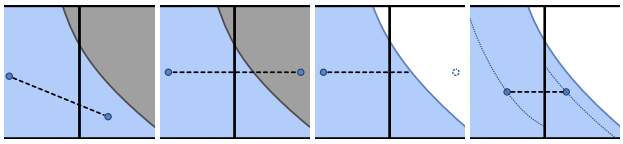
[1] Face vertices do not exist for 2-D.
[2] Grid and geometry edges could also be represented in 3-D, but these structures are not crucial for the method's explanation.

Laplacian as

$$\int_{\Omega_G} \nabla p \cdot \mathbf{n}\, dS + \int_{\Omega_B} \nabla p \cdot \mathbf{n}\, dS = \frac{\rho}{\Delta t} \int_{\Omega_G} \mathbf{u}^* \cdot \mathbf{n}\, dS + \frac{\rho}{\Delta t} \int_{\Omega_B} \mathbf{u}^* \cdot \mathbf{n}\, dS, \quad (1)$$

where p denotes the pressure field enforcing incompressibility, $\mathbf{n}$ is the cell's boundary normal, $\mathbf{u}^*$ is the velocity field after advection, ρ and Δt are the fluid density and time-step size, respectively. Notice that we distinguish between integrals discretized over grid aligned boundaries (Ω_G) and non-grid aligned boundaries (Ω_B).

Pressure gradients integrated along an immersed solid boundary can be factored out using the free-slip condition. Thus, the second term on the left-hand side of Equation (1) is omitted when Ω_B represents solid boundaries. However, discrete pressure gradients across internal regular faces Ω_G still have to be accounted for. Placing pressure samples at the centroids of cut-cells will result in gradients that are non-orthogonal relative to cell boundary normals (Figure 5 (a)) and require quadratic interpolation to keep the finite volume discretization consistent [Johansen and Colella 1998; Schwartz et al. 2006]. The orthogonality relationship between the discrete operators is then lost and the system is not symmetric positive definite anymore [Batty 2017]. Thus, previous works [Azevedo et al. 2016; Ng et al. 2009] repositioned pressure samples to lie on the centroids of regular grids (Figure 5 (b)). Conveniently, this simplification still maintains a second-order convergence for pressure values and first-order for gradients, even in cases where pressure locations are located inside solid obstacles.



(a) CC Centroids **(b)** GC Centroids **(c)** GFM **(d)** Ours

Fig. 5. Possible pressure samples placements for modelling immersed boundaries. Blue cells represent the fluid phase, grey cells are solid boundaries and white cells represent the air phase. Pressure samples at centroids of cut-cells [Johansen and Colella 1998] (a); pressure samples at centroids of the regular grid cell [Azevedo et al. 2016] (b); GFM enforces zero pressure at the surface, discarding cut-cell pressure samples outside of the liquid (c); our method positions pressure samples at iso-values (d).

In order to demonstrate clear intuitive illustrations, we will restrict discretizations derivations to 2-D. Since the solid boundaries, and subsequently cut-cells, are assumed to be piece-wise linear, we can discretize integrals in Equation (1) by assuming the integrated values to be constant for each edge. Hence, Equation (1) is discretized for the i-th cell as

$$\sum_{E_j \in \Omega^i_G} \|E_j\| \, (p_i - p_j) = \frac{-\rho \Delta x}{\Delta t} \sum_{E \in \Omega^i} \|E_j\| \, (\mathbf{u}^*_{E_j} \cdot \mathbf{n}_{E_j}) \quad (2)$$

where $\Omega^i = \Omega^i_G \cup \Omega^i_B$, E_j and $||E_j||$ are the j-th edge and its length, p_i is the i-th cell pressure value, p_j is the neighboring cell pressure value connected to the j-th cut-cell grid edge, and $\mathbf{u}^*_{E_j}$ and $\mathbf{n}_{E_j}$ are the discrete post-advection velocities and normals associated with the edge. The symbols of Equation (2) are visualized in Figure 4. The resulting linear system (Table 1 in [Azevedo et al. 2016]) is symmetric positive definite and can be readily solved by Conjugate Gradient algorithms.

2.1.3 Boundary Respecting Interpolants. Besides storing edge (2-D) or face (3-D) velocities, cut-cells also make use of vertex velocities. A vertex velocity is obtained after averaging cut-cells components connected to it, usually after the pressure projection step. Representing velocities on vertices allows the use of Spherical Barycentric Coordinates (SBC) [Langer et al. 2006] to interpolate the velocities for evaluating the advection step. The SBC is a boundary respecting interpolant, which can correctly model velocity profiles close to interfaces, generating path-lines that do not cross object geometries. For further implementation details on how to compute free-slip and no-slip velocities, we refer to [Azevedo et al. 2016].

3 AN EXTENDED CUT-CELL METHOD FOR SOLID AND LIQUID EMBEDDED BOUNDARIES

The Hybrid FLIP method [Zhu and Bridson 2005] employs particles to solve the advection step and to represent liquid surfaces, while incompressibility is enforced on the grid. By sampling several particles per-cell, the FLIP solver is able to track implicit surfaces with a higher resolution than the pressure solver itself. It is a common practice to calculate grid-based signed distance functions from the particle positions, and explicit liquid meshes are usually obtained by marching cubes. We assume that the air phase is neglected ($p = 0$) during the simulation, since a great portion of the visible phenomena happens because of the liquid movement itself. This assumption requires extrapolation of velocities from the liquid to the air phase so particles can be correctly transported during advection. The described simulation setup will yield liquid surfaces that are not aligned with the grid and hence need special treatment.

The Ghost Fluid Method (GFM) [Fedkiw et al. 1999] provides efficient embedding of non-conforming liquid surfaces inside regular grids. It avoids voxelized pressure artifacts by placing ghost values outside liquids and inside the air phase for each discretized gradient. The ghost pressure values are computed to enforce the zero pressure condition at the exact surface location, extrapolating pressures linearly towards the liquid-air interface. Boundary conditions are discretized with symmetric positive definite Laplacian matrices, which exhibit second-order accuracy for values and first-order for gradients [Fedkiw et al. 1999]. The GFM is widely adopted in computer graphics pipelines due to its efficiency, robustness, convergence and simplicity of implementation.

However, the Ghost Fluid Method cannot capture details smaller than a grid cell. If a cell center is *outside* the liquid surface, the GFM automatically tags it as an air cell (Figure 5 (c)), and the solver incorrectly ignores water regions outside cell centers. The mismatch between the pressure degrees of freedom represented by the grid and the discrete surface will result in missing details or permanent artifacts, especially for droplets pinching-off the surface, thin sheet splashes, pockets of air entrapped by the liquid, and narrow gaps in the surrounding solid obstacles. Additionally, velocities are extrapolated indistinguishably to the air or solid regions, violating

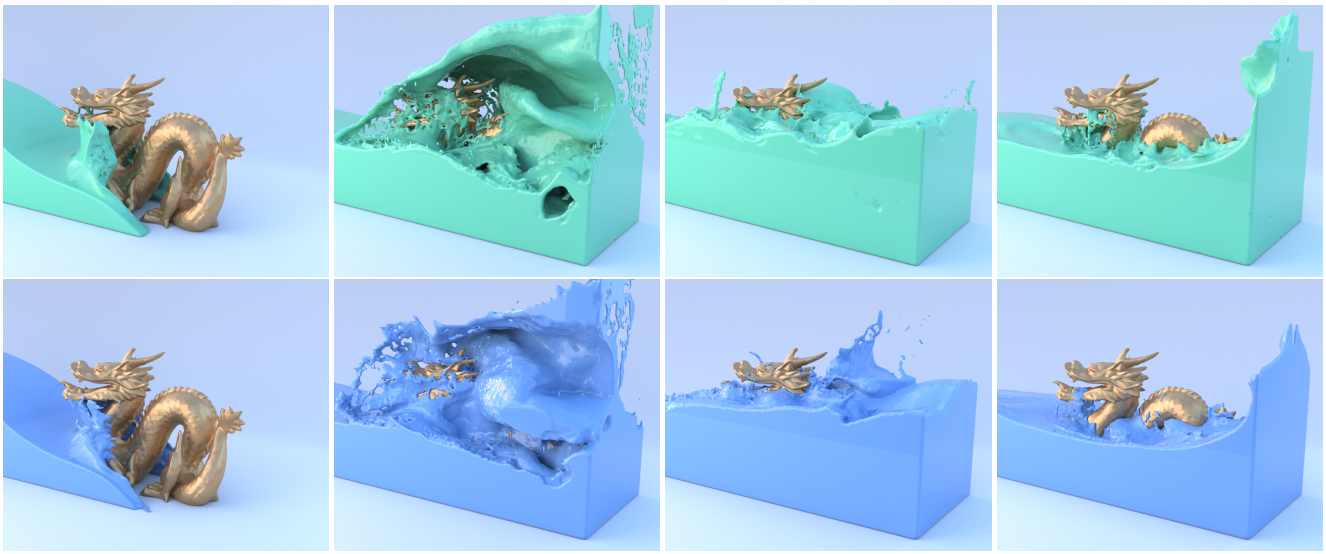


Fig. 6. Dragon in a liquid simulation. Top: GFM, bottom: ours. Our method reliably preserves the geometry at the solid-liquid interface, preventing voxelization artifacts near the dragon.

incompressibility. Unless explicitly solved with another pressure projection [Rasmussen et al. 2004], the extrapolation will deteriorate the volume of fluids in areas close to free surfaces or embedded obstacles.

3.1 A Novel Iso-Surface Poisson Solver

Our goal is to extend previous cut-cells methods to handle liquid interface tracking using a higher resolution than the pressure solver itself. For doing so, we want to keep track of all sub-grid liquid pressure samples stored on cut-cells, yielding a topologically correct discretization. In contrast to solids, liquids impose Dirichlet conditions ($p = 0$), causing non-vanishing normal pressure gradients at the liquid-air interface. Therefore, the pressure solver needs to evaluate gradients across non-grid aligned boundaries, and the integral of Equation (1) over Ω_B does not vanish. To evaluate this pressure gradient integral, pressure samples inside the liquid are required. Hence, shifting pressures locations to centroids of regular cells - as previous cut-cell approaches - would not yield a valid discretization of the liquid-air pressure gradient.

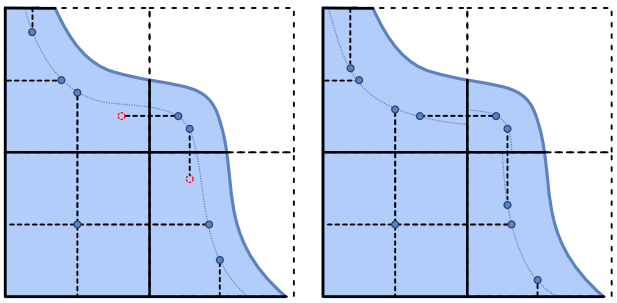
To solve these limitations, we developed a novel iso-surface Laplacian solver that correctly embeds non-grid aligned Dirichlet conditions while preserving symmetry and positive definiteness of the discrete operators. Our method derives from the intuition that without surface tension, the normal pressure gradient does not vary tangentially across the liquid interface (Figure 9). Thus, if we similarly to GFM assume that the pressures vary linearly along the normal direction of the liquid-air interface, we can find a per cut-cell iso-distance ϕ_i from the liquid surface where the pressure values are identical. This intuition allows us to reduce several pressure samples along the same iso-distance to a *single variable*, which results in a method that effectively stores a single pressure per cut-cell. Experiments shown in the supplementary material demonstrate that our method converges with second-order accuracy for values, and first-order accuracy for gradients.

Our iso-surface Laplacian solver models pressure samples that are always inside the liquid phase. This property enables the discretization of free-surface pressure gradients aligned with liquid-air interface normals, as further detailed in Section 3.3. Interestingly, our method represents a *single pressure* sample with *multiple physical locations*, depending on the gradient being evaluated, as shown in Figure 7. Notice that while the pressure positions change relatively inside a cut-cell, their gradients are still orthogonal to the cell boundary. This constraint, along with a linearly varying pressure profile at the normal direction of the liquid-air interface, guarantees the symmetry and positive definiteness of our discrete operators. The next Section details how iso-distances are computed per cut-cell, and in Section 3.3 we demonstrate how these values are used to discretize the pressure gradients.

3.2 Computing Cut-Cell Iso-Distances

Our solver defines discontinuous iso-distance values per cut-cell in order to discretize pressure gradients, which is enabled by pressure samples inside a same cut-cell referring to the same iso-distance. While one could simply adopt a single iso-distance for the whole liquid surface, that would hinder the discretization accuracy. A single iso-distance assumption positions pressure gradient centroids disregarding discrete velocity sample positions, which are assumed to lie on the centroid of grid edges (2-D) or grid faces (3-D). In Figure 7, we compare discretizations between continuous (a) and discontinuous (b) iso-distance values.

In staggered grids, matching the velocity and pressure gradient positions enhances the stability of the method [Harlow and Welch 1965]. Thus, it is desirable that all iso-distance values defined per



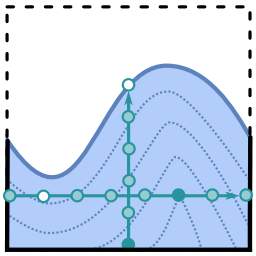
(a) Uniform iso-distances (b) Discontinuous iso-distances

Fig. 7. Placement of pressure samples across different cells: in (a), if a single iso-surface is used for every cell, certain samples cannot be properly placed (red dashed circles), compromising the accuracy of the pressures; in (b), varying the iso-distance between cells brings more flexibility to the placement of the pressure samples.

cut-cell minimize

$$\begin{aligned} \arg\min_{\phi_i} \sum_{i}^{\text{cells}} \sum_{j}^{\text{edges}} ||\mathbf{x}^{\nabla p_j}(\phi_i) - \mathbf{x}^{\mathbf{u}_j}||_2^2 \\ \textbf{s.t. } \phi_i^{\min} < \phi_i < 0, \end{aligned} \tag{3}$$

where ϕ_i is the iso-distance of the i-th cell, $\mathbf{x}^{\nabla p_j}(\phi_i)$ is the center position of the j-th pressure gradient normal to the cell boundary, $\mathbf{x}^{\mathbf{u}_j}$ the position of the staggered velocity, and $\phi_i^{\min}$ the minimum iso-distance value for the i-th cell. Changing ϕ_i values will shift pressure gradients' center locations, while velocity locations are kept fixed. Assuming that the signed distance value is negative inside the liquid, the iso-distance ϕ_i is constrained to positions that are both inside the liquid ($\phi_i < 0$) and the cut-cell ($\phi_i > \phi_i^{\min}$). The optimal solution of Equation (3) has to be computed considering all cut-cells simultaneously and the problem is over-constrained, since the number of variables and equations are equal to the number of cut-cells and edges, respectively.



Since the energy function of Equation (3) is smooth, one could solve it with an iterative gradient descent solver [Zhu et al. 1997]. However, solving this optimization at each time-step of the flow solver is computationally inefficient. Thus, we found that in practice, it is better to approximate the solution heuristically. We detail our heuristic as follows: The high-resolution liquid tracking grid is queried by a scan ray starting at grid edges centroids, following their normal direction inside the cut-cell. The inset image shows an example for two edges in 2D. Scan rays (turquoise arrows) access iso-distances on the fine-resolution level set grid, storing minimum (solid turquoise circle) and maximum values (white circles) as one interval. This interval keeps track of valid iso-distances that are aligned with the pressure gradient, and thus, are valid iso-distance candidates. After repeating this step for all grid edges of a single cut-cell, we define a new per cut-cell interval by discarding minimum and maximum ranges that do not satisfy all scan ray intervals computed individually per edge. This step minimizes the occurrence of scan rays not reaching valid iso-distances (Section 3.3). Finally, ϕ_i is set to the middle point of the intersection of the per cut-cell interval. While we do not guarantee the optimal solution, this heuristic locally minimizes the energy function while also satisfying the hard constraints imposed by the optimization problem. We provide an extended discussion and algorithmic details in the supplementary material, in which we show that the discontinuous approach is indeed more accurate than its continuous counterpart.

3.3 Discrete Pressure Gradients

We follow the same finite volume formulation as Equation (2), computing pressure gradients over grid (Ω_G) and geometry (Ω_B) edges (faces in 3-D). However, to compute Ω_G pressure gradients, we need to first find the new pressure locations, which depend on iso-distances computed per cut-cell. We again make use of scan ray searches, however this time finding the previously agreed iso-distance value per cut-cell. If the ray fails to find the iso-distance value, we use the centroid of the regular cell. Considering the j-th edge of the i-th cut-cell, two rays on opposite directions are cast, and the pressure gradient is approximated by

$$\int_{E_j} \nabla p \cdot \mathbf{n}\, dS \approx \frac{-\Delta x\, \|E_j\|\, (p_i - p_j)}{\|\mathbf{x}^{p_i} - \mathbf{x}^{p_j}\|} \; \forall E_j \in \Omega_G^i, \tag{4}$$

where $\mathbf{x}^{p_i}$ and $\mathbf{x}^{p_j}$ are the positions of the pressure samples of the i-th cell and its j-th pressure neighbor. The term Δx on the denominator is necessary to normalize the distance $||\mathbf{x}^{p_i} - \mathbf{x}^{p_j}||$ to the interval $[0, 1]$ in the same fashion as $\|E_j\|$. We notice that the distance between the iso-distance pressure location and the edge (face in 3-D) centroid may be arbitrarily small and compromise the numerical stability of the method. In this case, we simply set the pressure sample in which this happens to zero, since this it is already close enough to the liquid interface. We used this strategy for all examples shown in Section 5, discarding gradients in which the distance between the pressure location and the liquid-air interface is smaller than 0.1% of the simulation grid size.

The pressure gradient along the liquid-air interface can be similarly discretized, since in the absence of surface tension, the pressure at the surface is zero. Using finite differences between pressures that lie at the liquid surface and at the iso-distance results in

$$\int_{\Omega_A^i} \nabla p \cdot \mathbf{n}\, dS \approx \sum_{E_j \in \Omega_A^i} \frac{\Delta x\, \|E_j\|\, (p_i - 0)}{\phi_i}, \tag{5}$$

where Ω_A^i is the liquid–air boundary for the i-th cut cell. Combining Equations 4 and 5 yields a linear system for pressures that is symmetric positive definite. We can verify this by the following relation for the i-th cut-cell

$$\sum_{E_j \in \Omega_G^i} \frac{\|E_j\|}{\|\mathbf{x}^{p_i} - \mathbf{x}^{p_j}\|} - \sum_{E_j \in \Omega_A^i} \frac{\|E_j\|}{\phi_i} \geq \sum_{E_j \in \Omega_G^i} \frac{\|E_j\|}{\|\mathbf{x}^{p_i} - \mathbf{x}^{p_j}\|}, \tag{6}$$

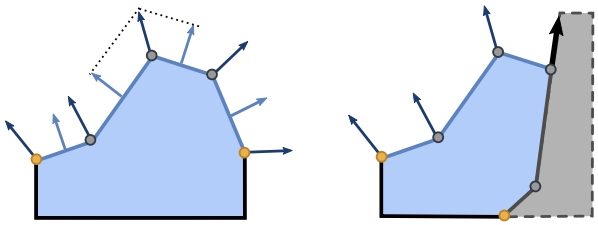
where the left-hand side is the sum of all coefficients of the p_i pressure (main diagonal) and the right-hand side is the sum of the coefficients of the p_j neighboring pressures. We highlight again the importance of having pressure samples inside the liquid phase:

this guarantees that ϕ_i values that are always negative, and thus, Equation (6) holds. Additionally, solid boundaries enforce Neumann conditions, which do not change the gradient stencils on the discrete Laplacian matrix described by our approach. Hence, solids immersed on liquids are automatically accounted for without modifying the previous equations.

Thus, the linear system for computing pressures without the presence of surface tension is

$$\sum_{E_j \in \Omega^i_G} \frac{\|E_j\| \, (p_i - p_j)}{\|\mathbf{x}^{p_i} - \mathbf{x}^{p_j}\|} - \sum_{E_j \in \Omega^i_A} \frac{\|E_j\| \, p_i}{\phi_i} = \frac{-\rho}{\Delta t} \sum_{E_j \in \Omega^i} \|E_j\| \, (\mathbf{u}^*_{E_j} \cdot \mathbf{n}_{E_j}). \quad (7)$$

The above equation requires per edge post-advection velocities $\mathbf{u}^*_{E_j}$. To compute those, our algorithm transfers FLIP particle velocities to all grid edges (faces in 3-D) and geometry vertices, similarly to [Azevedo et al. 2016]. Once the post-advection velocities are computed per vertex, they are averaged to become edge (face in 3-D) velocities, and can be used on the right-hand side of Equation (7).



(a) (b)

Fig. 8. Updating the liquid-air interface (a): For the edges (light blue arrows), we update in the normal direction. Vertices velocities (dark blue arrows) are interpolated such that they match the normal component of neighboring edges (black dashed lines). For vertices on a triple junction (b), the update direction is tangential to the solid surface (black arrow).

3.4 Updating Velocities to Enforce Incompressibility

After the pressure of each cell is known, we update the velocities to enforce incompressibility of the grid-aligned boundaries and liquid–air interfaces. For grid edges, this update uses the modified pressure sample positions in the discrete gradient as

$$\mathbf{u}^{n+1}_{E_j} = \mathbf{u}^*_{E_j} + \frac{\Delta t}{\rho} \frac{(p_i - p_j)}{\|\mathbf{x}^{p_i} - \mathbf{x}^{p_j}\|} \mathbf{n}_{E_j} \quad \forall E_j \in \Omega^i_G. \quad (8)$$

For edges on the liquid-air interface, the pressure gradient is normal to the liquid surface, hence we only update the velocity in the normal direction. While one could compute a discrete pressure gradient for liquid-air interfaces by using Equation (5), this method can suffer from instabilities when the computed cut-cell iso-distance is too small. We can exploit our knowledge of cut-cell geometry to enforce incompressibility in a discrete sense by a method in which we dub the *budget method*. In this method, a *flux budget* is computed by summing grid edge velocities after projection with post-advection geometry edge velocities by

$$F^b_i = \sum_{E_j \in \Omega^i_G} \|E_j\| \, (\mathbf{u}^{n+1}_{E_j} \cdot \mathbf{n}_{E_j}) + \sum_{E_j \in \Omega^i_B} \|E_j\| \, (\mathbf{u}^{n*}_{E_j} \cdot \mathbf{n}_{E_j}). \quad (9)$$

We can then redistribute this flux budget evenly to all liquid-air edges, yielding a discrete divergence $\sum_{E_j \in \Omega^i} \|E_j\| \, (\mathbf{u}^{n+1}_{E_j} \cdot \mathbf{n}_{E_j})$ that sums up exactly to be zero. That is, for each edge E_j we compute a correction to liquid-air post-advection velocities as

$$\mathbf{u}^{n+1}_{E_j} = \mathbf{u}^{n*}_{E_j} - \frac{F^b_i}{\sum_{E_j \in \Omega^i_A} \|E_j\|} \quad \forall E_j \in \Omega^i_A. \quad (10)$$

We notice that this equation might cause issues when only a tiny portion of the cut-cell is a liquid-air interface. Iterative pressure solvers are limited by their residual tolerance values, and thus the sum of the discrete divergence will not be exactly zero for each cell. This error will be amplified by the small liquid-air interface, causing the budget method velocities to overshoot. Thus, our solver clamps the magnitude of the correction to not be larger than $\left|\frac{\Delta t p_i}{\rho \phi_i}\right|$, which effectively relaxes incompressibility in favor of stability.

Once the per-edge post-projection velocities are computed, we need to accordingly update vertex velocities. While grid vertex velocities are simply averaged from the connected cut-cell structures, edge (face in 3-D) and geometry vertices require special attention. These vertices are connected to edges (faces in 3-D) which have normal-oriented fluxes. We employ a Least-Squares fitting strategy as [Azevedo et al. 2016; Klingner et al. 2006], which combines several normal contributions to form a full 2-D or 3-D velocity. This velocity is used to compute only the normal velocity of a vertex, since the tangential velocities are kept to be the same as the ones transferred by the FLIP particles. If the vertex is on a triple junction between the air, liquid and solid (Figure 8), we modify its normal to remain in tangent direction of the solid normal, so the correction does not create velocities that penetrate the solid obstacle.

4 SURFACE TENSION

As noticed by previous works [Sussman and Ohta 2009], surface tension forces are more stable if integrated by the mean curvature flow of the liquid interface as

$$\frac{\partial \mathbf{x}}{\partial t} = \sigma \nabla^2 \mathbf{x} \; \forall \mathbf{x} \in \Omega_A, \quad (11)$$

where σ is the surface tension value and Ω_A is the liquid-air interface. We use an extension of the original method which runs directly on triangle meshes, proposed by [Thürey et al. 2010]. The liquid-air interface is first updated implicitly by solving Equation (11). Then, the difference between the updated and the original mesh positions are used to compute pressure values at the original mesh.

In the presence of surface tension, the surface pressure is no longer constant and the pressure iso-surfaces do not align with the signed distance field (Figure 9). Furthermore, the change in pressure also limits our knowledge about values underneath the surface, making it challenging to relocate the pressure samples such that they share one value in a cell. Thus, Equation (7) is no longer sufficient for modelling pressure values at the iso-surface. We notice

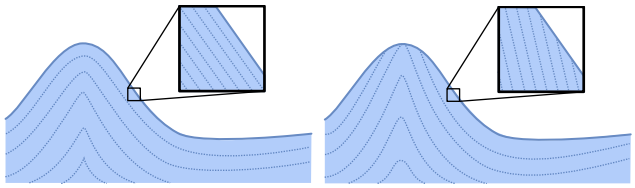


Fig. 9. Without surface tension, pressure iso-values are parallel to the free surface (left). With surface tension (right), the pressure iso-surfaces do not align with the surface and the iso-surface assumption no longer holds. Given a small enough region, both pressure fields can be approximated with linear functions, and our assumption of pressure gradients on the surface becomes a good approximation.

that our iso-surface Laplacian solver models pressures as functions that vary linearly along the liquid-air interface normal directions. A pressure sample $\hat{p}_i$ can be written in terms of the pressure value of closest point at the liquid interface as

$$\hat{p}_i = a_i \phi_i + q \tag{12}$$

where a_i is the rate in which the pressure gradient varies away from the liquid-air interface, and q is the pressure value at closest point on the surface. Notice that the original pressures relate to $\hat{p}_i$ as $p_i = a_i \phi_i$.

Directly replacing this linear relation into Equation (7) will yield a coupled system for the pressures at the surface and the ones inside the liquid. Solving this coupled system may induce instability issues due fast travelling waves at the liquid surface [Thürey et al. 2010]. Thus, instead of solving for the pressure $\hat{p}_i$ directly, we decouple the solution by first solving for q at the liquid mesh using Equation (11). Once surface pressures are computed, these can be added to the cut-cell pressures and will conveniently vanish in the liquid-air interface normal direction as

$$\begin{aligned}\int_{\Omega^i_A} \nabla p \cdot \mathbf{n} &\approx \sum_{E_j \in \Omega^i_A} \frac{\|E_j\| \, (\hat{p}_i - q)}{\phi_i} \\ &= \sum_{E_j \in \Omega^i_A} \frac{\|E_j\| \, (p_i + q - q)}{\phi_i} = \sum_{E_j \in \Omega^i_A} \frac{\|E_j\| \, p_i}{\phi_i}. \end{aligned} \tag{13}$$

In Figure 10, we show that a q variable is added for every pressure sample point used on the discretization of gradients. For the j-th edge of the i-th cell, samples q^j_i and q^i_j are added inside and outside the cut-cell, respectively. Substituting these in Equation (7) and moving the q variables to the right-hand side yields the following system

$$\begin{aligned}&\sum_{E_j \in \Omega^i_G} \frac{\|E_j\| \, (p_i - p_j)}{\|\mathbf{x}^{p_i} - \mathbf{x}^{p_j}\|} - \sum_{E_j \in \Omega^i_A} \frac{\|E_j\| \, p_i}{\phi_i} \\ &= - \sum_{E_j \in \Omega^i_G} \frac{\|E_j\| \, \left(q^j_i - q^i_j\right)}{\|\mathbf{x}^{p_i} - \mathbf{x}^{p_j}\|} - \frac{\rho}{\Delta t} \sum_{E_j \in \Omega^i} \|E_j\| \, (\mathbf{u}^*_{E_j} \cdot \mathbf{n}_{E_j}). \end{aligned} \tag{14}$$

This equation only differs from Equation (7) by an extra term on the right-hand side. Once pressures are computed with the above equations, we can then use the surface tension modified pressures to update the grid face velocities with Equation (8). To update the surface velocity with surface tension, an extra tangential pressure gradient, which can be calculated for each vertex with neighboring vertices, is added to the vertices. Note that for velocity updates in the normal direction, Equation (13) is exactly identical in form to Equation (5). This means that if we use pressure modified by the surface tension, the same steps in Section 3.4 can be used.

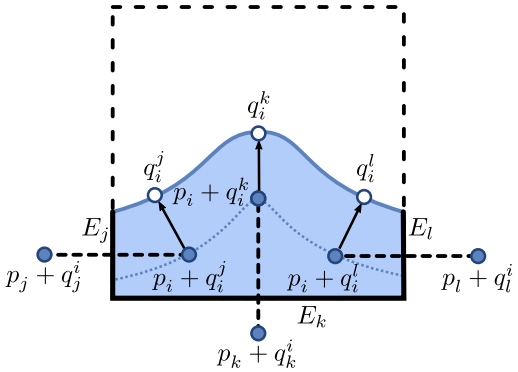


Fig. 10. To solve fluids with surface tension, an extra value is stored at each pressure sample. Note that the q term can vary within the cut cell.

Algorithm 1: Extended Liquid Cut-Cells Pipeline

```
Transport particles and update liquid position
Generate cut-cell meshes
Splat particle velocities to the vertices of the mesh and grid
Compute Iso-Distances per Cut-cell
if surface tension then
    Solve for q on liquid surface [Thürey et al. 2010]
Interpolate per-edge (face 3-D) velocities from vertices
Solve pressures with Equation (14)
Project grid edge (faces 3-D) velocities
if surface tension then
    Update tangential velocity of liquid–air interface
Compute budget method
Compute free surface edge (faces 3-D) velocities
Interpolate per-vertex velocities from edges (faces 3-D)
Update particle velocities with SBC
```

5 RESULTS

5.1 Implementation

Algorithm 1 summarizes the steps to perform one time-step with our method. The simulations were run on a standard desktop with an Intel i7-7700K CPU with 32 gigabytes of RAM, and the Poisson solver was implemented in CUDA and ran on an NVIDIA GeForce GTX 1080Ti. Table (1) summarizes our timing results of our extended cut-cells, while also depicting the resolution for tracking liquid

surfaces. We refer the reader to the accompanying video for better visualization of the presented results.

5.2 Simulation Results

Comparisons with GFM. In Figure 2, we show a side-by-side comparison between GFM and our extended cut-cells method. Three drops of varying sizes are initialized above a liquid basin, splashing as they fall into the liquid below. This example is discretized with varying grid resolutions ($16\times8\times12$, $32\times16\times24$, $64\times32\times48$, from left to right). As expected, at coarse resolutions, GFM is unable to capture small droplets, resulting in perceptible hanging artifacts. Our extended cut-cell method works even for coarse setups and small droplets, only suffering from dissipation when the grid resolution is insufficient.

Interaction with complex objects. Figure 6 shows a dragon mesh combined with a dam-break scene. Our extended cut-cell method is able to faithfully preserve the geometry at the solid–liquid interface, preventing voxelization artifacts near the dragon. Fluid further away from the dragon, however, behaves similarly for both methods. Figure 11 shows a water drop colliding with the dragon mesh and breaking into a thin sheet on impact. In this example, the level set grid is much finer than the simulation grid, and severe voxelization artifacts are noticeable near the obstacle with GFM. Furthermore, thin liquid sheets, which make up most of the simulation, are assumed to have zero pressure, leading to incompressibility not being properly maintained, which is especially apparent at the domain boundaries. When such sheets are represented with cut-cells, meaningful pressure values are calculated to ensure incompressibility, producing results that look more natural. Figure 3 features a liquid flowing out of an array of narrow slits. In GFM, the slits can only be detected when they are directly aligned with the grid cells, leading them to be erroneously simulated as fewer, wider gaps in the wall. Our method, on the other hand, manages to detect these gaps, allowing the liquid to flow through all slits. Due to the coarseness of the simulation, artifacts due to extrapolation also become apparent with GFM, manifesting as floating blobs in the simulation.

Surface tension. We validate our surface tension approach in Figure 12, which shows a simple scene with a cube in zero gravity with a surface tension value of $\sigma = 5\times10^{-4}$. The liquid attempts to minimize its own surface area, gradually becoming rounder and causing surface waves in the process. In a similar but more complex scenario, Figure 13 shows the same setup, this time with an armadillo-shaped liquid. The left most picture in Figure 1 depicts a liquid splash in shallow water with surface tension value $\sigma = 2\times10^{-4}$. Finally, Figure 14 simulates a water drop breaking into smaller droplets upon impact with a spike. Droplet break-up into sub-cell elements interacting with such complex obstacles is particularly challenging to capture with traditional approaches.

6 CONCLUSIONS AND FUTURE WORK

In this work, we extended the cut-cells method [Azevedo et al. 2016] to include free-surface boundary conditions and surface tension. This addition allows intricate simulations of surface phenomena at a sub-grid resolution. Furthermore, due to the topological awareness of cut-cells, surface tension can be incorporated elegantly into the simulation, allowing potential for complex simulations at a lower grid resolution. In terms of the overall liquid results, we demonstrated that our method shows a clear advantage over traditional GFM in terms of simulation precision and versatility. Solid surface topology, as well as detailed meshes and complex topological structures, can be properly captured and simulated without requiring unreasonable grid resolutions. Numerical experiments in our supplemental material also show that our novel Poisson solver achieves second-order accuracy for the pressure values and first-order for pressure gradients.

However, our method is a first attempt at capturing liquids with cut-cells, and some limitations are also present in our results. Firstly, due to the varying sizes of the cut-cells, it is known for FLIP to be unstable around solid boundaries [Azevedo et al. 2016] and PIC must be used in such cases. This leads to some dissipation around the solid boundaries. Recent advances in advection, such as APIC [Jiang et al. 2015] and PolyPIC [Fu et al. 2017] could potentially be used to solve the instability and dissipation. Due to the implicit way we derive surface tension, our simulations exhibit greater stability than directly finding the surface pressure using the mean curvature. However, some surface tension phenomena cannot be properly simulated when larger time-steps are taken. For example, Figure 14 requires a small time-step in order to correctly simulate the breaking-up of the droplets. Furthermore, the discrepancy of resolution between the simulation grid and the level set means that a smaller time-step has to be used than if no discrepancy existed. [Thürey et al. 2010] addresses these issues by evolving the mesh via smoothing or simulating surface waves to achieve a stable approximation of surface tension. This is currently not feasible in our method, since modifying the mesh can cause particles to be outside the liquid surface, causing wrong velocities to be interpolated in such cases.

While this was not the focus of this work, the simulation time could be greatly improved. Table 2 shows the running time for each individual step of our method. Our cut-cells implementation is not optimized for speed, thus meshes are slow to generate and update. We believe that using the state-of-the-art cut-cells framework [Tao et al. 2019] would greatly improve the performance of our method. The use of an explicit mesh also has additional disadvantages, such as various numerical instabilities, as well as being subject to biases in the surface extraction, such as the curling boundaries in the splash of Figure 1. We notice that computing iso-values and pressure sample positions has little impact on the overall time spent per frame. Thus, a potential way to improve on the current method is to directly use the level set to infer connectivity in a grid cell, employing our cut-cells only as a high-level concept for adaptive simulation. Other directions for further research include incorporating alternative simulation techniques, such as the use of stream functions [Ando et al. 2015] or modelling air bubbles [Goldade and Batty 2017]. Nevertheless, we believe that our method laid down a solid foundation for future work in more precise and adaptive liquid representations.

REFERENCES

Mridul Aanjaneya, Ming Gao, Haixiang Liu, Christopher Batty, and Eftychios Sifakis. 2017. Power diagrams and sparse paged grids for high resolution adaptive liquids.

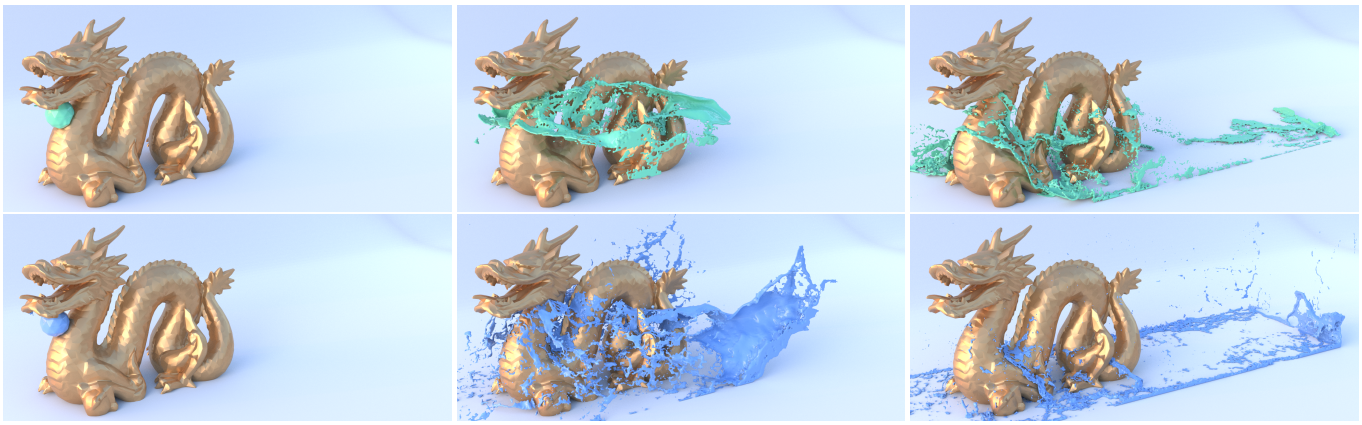


Fig. 11. A sideways travelling liquid sphere breaking into a sheet on impact. Even with level sets that are much finer than the grid resolution used by the solver, our method is still able to correctly simulate thin sheets.

Table 1. Average running time per frame (in seconds) of our simulations for GFM and our method (CC). Here, "Simulation" denotes the simulation grid size, while "Tracker" denotes the surface tracker resolution.

	Simulation	Tracker	GFM	CC	#Frames	Time-steps per frame
Drops (low)	$16 \times 8 \times 12$	$128 \times 64 \times 96$	1.561	11.289	180	1
Drops (mid)	$32 \times 16 \times 24$	$128 \times 64 \times 96$	2.478	7.161	180	1
Drops (high)	$64 \times 32 \times 48$	$128 \times 64 \times 96$	2.438	7.244	180	1
Dragon	$128 \times 64 \times 32$	$512 \times 256 \times 128$	57.904	102.731	510	1
Drop	$96 \times 32 \times 32$	$768 \times 256 \times 256$	16.878	77.018	327	1
Narrow slits	$64 \times 64 \times 32$	$256 \times 256 \times 128$	39.529	89.990	480	1
Cube	$100 \times 100 \times 100$	$200 \times 200 \times 200$	–	11.824	1500	1
Armadillo	$100 \times 100 \times 100$	$200 \times 200 \times 200$	–	11.589	450	2
Spike	$50 \times 64 \times 50$	$200 \times 256 \times 200$	–	8.144	200	5
Splash	$102 \times 102 \times 102$	$408 \times 408 \times 408$	–	263.555	625	1

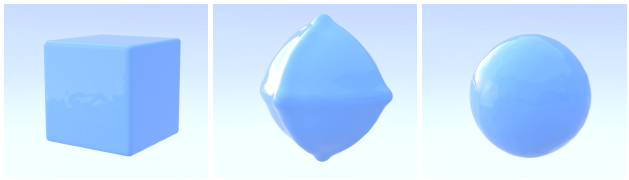


Fig. 12. A liquid cube oscillates in zero gravity under surface tension with surface tension value $\sigma = 5 \times 10^{-4}$. As expected, the fluid minimizes its surface area and gradually becomes a sphere.

Fig. 13. A liquid armadillo in zero gravity with surface tension value $\sigma = 5 \times 10^{-4}$.

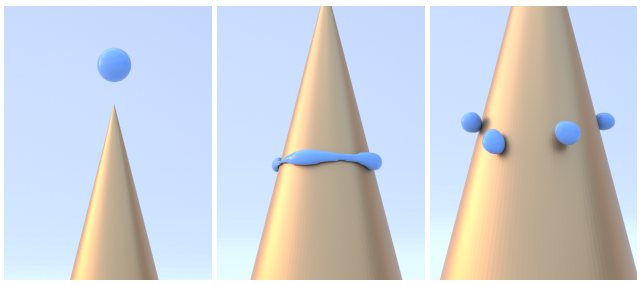


Fig. 14. Droplet with a larger surface tension value ($\sigma = 4 \times 10^{-3}$) breaking into smaller droplets on a spike.

ACM Transactions on Graphics 36, 4 (jul 2017), 1–12. https://doi.org/10.1145/3072959.3073625

Ryoichi Ando, N. Thurey, and R. Tsuruno. 2012. Preserving Fluid Sheets with Adaptively Sampled Anisotropic Particles. *IEEE Transactions on Visualization and Computer Graphics* 18, 8 (aug 2012), 1202–1214. https://doi.org/10.1109/TVCG.2012.87

Ryoichi Ando, Nils Thürey, and Chris Wojtan. 2013. Highly adaptive liquid simulations on tetrahedral meshes. *ACM Transactions on Graphics* 32, 4 (jul 2013), 1. https://doi.org/10.1145/2461912.2461982

Ryoichi Ando, Nils Thürey, and Chris Wojtan. 2015. A Dimension-Reduced Pressure Solver for Liquid Simulations. *Comput. Graph. Forum* 34, 2 (May 2015), 473–480.

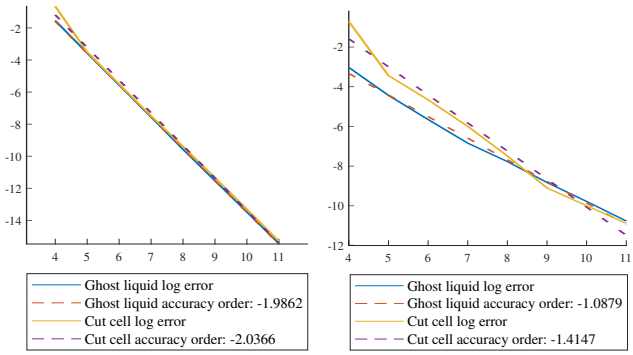


Fig. 15. Reconstruction error of the values (left) and gradients (right) of a function ("Peaks" in the supplementary paper) with Dirchlet boundary conditions. Our method performs similarly to GFM.

Table 2. Detailed running time of our cut-cell solver sub-steps for the Dragon example (Figure 6). Steps marked in bold differ from GFM or are exclusive to our method. We chose to measure the time for the frame 114 of the simulation, which exhibits large amounts of splashing and thin liquid sheets. The presence of detailed liquid surfaces results in slower cut-cells generation performance.

Step	Time (secs)
Particle transportation	3.86
Level set and marching cubes	48.486
Cut-cell generation	70.671
Particle velocity to grid vertices	7.31
Particle velocity to cut-cell vertices	18.5
Signed distance field update	8.546
Iso-value and pressure sample locations	0.254
Solving Poisson system	1.452
Grid face pressure projection	0.133
Liquid–air interface velocity update	3.445
Liquid–solid interface velocity update	1.328
Grid to particle velocity transfer	10.809
Total	174.794

https://doi.org/10.1111/cgf.12576
Vinicius C Azevedo, Christopher Batty, and Manuel M Oliveira. 2016. Preserving Geometry and Topology for Fluid Flows with Thin Obstacles and Narrow Gaps. *ACM Trans. Graph.* 35, 4 (2016), 97:1—-97:12. https://doi.org/10.1145/2897824.2925919
Adam W. Bargteil, Tolga G. Goktekin, James F. O'brien, and John A. Strain. 2006. A semi-Lagrangian contouring method for fluid simulation. *ACM Transactions on Graphics* 25, 1 (jan 2006), 19–38. https://doi.org/10.1145/1122501.1122503
Christopher Batty. 2017. A cell-centred finite volume method for the Poisson problem on non-graded quadtrees with second order accurate gradients. *J. Comput. Phys.* 331 (feb 2017), 49–72. https://doi.org/10.1016/j.jcp.2016.11.035
Christopher Batty, Florence Bertails, and Robert Bridson. 2007. A fast variational framework for accurate solid-fluid coupling. *ACM Transactions on Graphics* 26, 3 (jul 2007), 100. https://doi.org/10.1145/1276377.1276502
Christopher Batty and Ben Houston. 2011. A Simple Finite Volume Method for Adaptive Viscous Liquids. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '11)*. ACM, New York, NY, USA, 111–118. https://doi.org/10.1145/2019406.2019421
Christopher Batty, Andres Uribe, Basile Audoly, and Eitan Grinspun. 2012. Discrete viscous sheets. *ACM Transactions on Graphics* 31, 4 (jul 2012), 1–7. https://doi.org/10.1145/2185520.2185609
Christopher Batty, Stefan Xenos, and Ben Houston. 2010. Tetrahedral Embedded Boundary Methods for Accurate and Flexible Adaptive Fluids. *Computer Graphics Forum* 29, 2 (may 2010), 695–704. https://doi.org/10.1111/j.1467-8659.2009.01639.x
Jacob Bedrossian, James H. [von Brecht], Siwei Zhu, Eftychios Sifakis, and Joseph M. Teran. 2010. A second order virtual node method for elliptic problems with interfaces and irregular domains. *J. Comput. Phys.* 229, 18 (2010), 6405 – 6426. https://doi.org/10.1016/j.jcp.2010.05.002
Morten Bojsen-Hansen and Chris Wojtan. 2013. Liquid surface tracking with error compensation. *ACM Transactions on Graphics* 32, 4 (jul 2013), 1. https://doi.org/10.1145/2461912.2461991
Tyson Brochu, Christopher Batty, and Robert Bridson. 2010. Matching fluid simulation elements to surface geometry and topology. *ACM Transactions on Graphics* 29, 4 (jul 2010), 1. https://doi.org/10.1145/1778765.1778784
Nuttapong Chentanez, Bryan E Feldman, François Labelle, James F O'Brien, James F O'Brien, and Jonathan R Shewchuk. 2007. Liquid Simulation on Lattice-based Tetrahedral Meshes. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '07)*. Eurographics Association, Goslar Germany, Germany, 219–228. http://dl.acm.org/citation.cfm?id=1272690.1272720
Paul W. Cleary, Soon Hyoung Pyo, Mahesh Prakash, and Bon Ki Koo. 2007. Bubbling and frothing liquids. *ACM Transactions on Graphics* 26, 3 (jul 2007). https://doi.org/10.1145/1276377.1276499
Jonathan M. Cohen and M. Jeroen Molemaker. 2004. Practical simulation of surface tension flows. In *ACM SIGGRAPH 2004 Sketches on - SIGGRAPH '04*. ACM Press, New York, New York, USA, 70. https://doi.org/10.1145/1186223.1186311
Fang Da, Christopher Batty, Chris Wojtan, and Eitan Grinspun. 2015. Double Bubbles sans Toil and Trouble: Discrete Circulation-Preserving Vortex Sheets for Soap Films and Foams. *ACM Trans. Graph.* 34, 4 (jul 2015). https://doi.org/10.1145/2767003
Fang Da, David Hahn, Christopher Batty, Chris Wojtan, and Eitan Grinspun. 2016. Surface-Only Liquids. *ACM Trans. Graph.* 35, 4 (jul 2016). https://doi.org/10.1145/2897824.2925899
Essex Edwards and Robert Bridson. 2014. Detailed water with coarse grids. *ACM Transactions on Graphics* 33, 4 (jul 2014), 1–9. https://doi.org/10.1145/2601097.2601167
R. Elliot English, Linhai Qiu, Yue Yu, and Ronald Fedkiw. 2013. Chimera grids for water simulation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation - SCA '13*. ACM Press, New York, New York, USA, 85. https://doi.org/10.1145/2485895.2485897
Douglas Enright, Stephen Marschner, and Ronald Fedkiw. 2002. Animation and rendering of complex water surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques - SIGGRAPH '02*. ACM Press, New York, New York, USA, 736. https://doi.org/10.1145/566570.566645
Ronald P Fedkiw, Tariq Aslam, Barry Merriman, and Stanley Osher. 1999. A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method). *J. Comput. Phys.* 152, 2 (jul 1999), 457–492. https://doi.org/10.1006/jcph.1999.6236
Nick Foster and Ronald Fedkiw. 2001. Practical animation of liquids. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01*. ACM Press, New York, New York, USA, 23–30. https://doi.org/10.1145/383259.383261
Nick Foster and Dimitri Metaxas. 1996. Realistic Animation of Liquids. *Graphical Models and Image Processing* 58, 5 (sep 1996), 471–483. https://doi.org/10.1006/gmip.1996.0039
Ben Frost, Alexey Stomakhin, and Hiroaki Narita. 2017. Moana. In *ACM SIGGRAPH 2017 Talks on - SIGGRAPH '17*. ACM Press, New York, New York, USA, 1–2. https://doi.org/10.1145/3084363.3085091
Chuyuan Fu, Qi Guo, Theodore Gast, Chenfanfu Jiang, and Joseph Teran. 2017. A Polynomial Particle-in-Cell Method. *ACM Trans. Graph.* 36, 6, Article 222 (Nov. 2017), 12 pages. https://doi.org/10.1145/3130800.3130878
Tolga G Goktekin, Adam W Bargteil, and James F O'Brien. 2004. A Method for Animating Viscoelastic Fluids. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2004)* 23, 3 (2004), 463–468. http://graphics.cs.berkeley.edu/papers/Goktekin-AMF-2004-08/
Ryan Goldade and Christopher Batty. 2017. Constraint Bubbles: Adding Efficient Zero-Density Bubbles to Incompressible Free Surface Flow. (11 2017).
Ryan Goldade, Christopher Batty, and Chris Wojtan. 2016. A Practical Method for High-Resolution Embedded Liquid Surfaces. *Computer Graphics Forum* 35, 2 (may 2016), 233–242. https://doi.org/10.1111/cgf.12826
Ryan Goldade, Yipeng Wang, Mridul Aanjaneya, and Christopher Batty. 2019. An adaptive variational finite difference framework for efficient symmetric octree viscosity. *ACM Transactions on Graphics* 38, 4 (jul 2019), 1–14. https://doi.org/10.1145/3306346.3322939
Francis H Harlow and J Eddie Welch. 1965. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The physics of fluids* 8, 12 (1965), 2182–2189.

Jeffrey Lee Hellrung, Luming Wang, Eftychios Sifakis, and Joseph M. Teran. 2012. A Second Order Virtual Node Method for Elliptic Problems with Interfaces and Irregular Domains in Three Dimensions. *J. Comput. Phys.* 231, 4 (Feb. 2012), 2015–2048. https://doi.org/10.1016/j.jcp.2011.11.023

Nambin Heo and Hyeong-Seok Ko. 2010. Detail-preserving fully-Eulerian interface tracking framework. *ACM Transactions on Graphics* 29, 6 (dec 2010), 1. https://doi.org/10.1145/1882261.1866198

Jeong-Mo Hong and Chang-Hun Kim. 2005. Discontinuous fluids. *ACM Transactions on Graphics* 24, 3 (jul 2005), 915. https://doi.org/10.1145/1073204.1073283

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The affine particle-in-cell method. *ACM Transactions on Graphics* 34, 4 (jul 2015), 51:1–51:10. https://doi.org/10.1145/2766996

Hans Johansen and Phillip Colella. 1998. A Cartesian Grid Embedded Boundary Method for Poisson's Equation on Irregular Domains. *J. Comput. Phys.* 147, 1 (nov 1998), 60–85. https://doi.org/10.1006/jcph.1998.5965

Myungjoo Kang, Ronald P. Fedkiw, and Xu-Dong Liu. 2000. A Boundary Condition Capturing Method for Multiphase Incompressible Flow. *Journal of Scientific Computing* 15, 3 (2000), 323–360. https://doi.org/10.1023/A:1011178417620

Byungmoon Kim, Yingjie Liu, Ignacio Llamas, Xiangmin Jiao, and Jarek Rossignac. 2007. Simulation of bubbles in foam with the volume control method. *ACM Transactions on Graphics* 26, 3 (jul 2007), 98. https://doi.org/10.1145/1276377.1276500

Doyub Kim, Oh-young Song, and Hyeong-Seok Ko. 2009. Stretching and wiggling liquids. *ACM Transactions on Graphics* 28, 5 (dec 2009), 1. https://doi.org/10.1145/1618452.1618466

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

Bryan M. Klingner, Bryan E. Feldman, Nuttapong Chentanez, and James F. O'Brien. 2006. Fluid animation with dynamic meshes. *ACM Transactions on Graphics* 25, 3 (jul 2006), 820. https://doi.org/10.1145/1141911.1141961

Torsten Langer, Alexander Belyaev, and Hans-Peter Seidel. 2006. Spherical Barycentric Coordinates. In *Symposium on Geometry Processing*, Alla Sheffer and Konrad Polthier (Eds.). The Eurographics Association. https://doi.org/10.2312/SGP/SGP06/081-088

Minjae Lee, David Hyde, Kevin Li, and Ronald Fedkiw. 2019. A Robust Volume Conserving Method for Character-Water Interaction. (feb 2019). arXiv:1902.00801 http://arxiv.org/abs/1902.00801

Frank Losasso, Frédéric Gibou, and Ron Fedkiw. 2004. Simulating water and smoke with an octree data structure. *ACM Transactions on Graphics* 23, 3 (aug 2004), 457. https://doi.org/10.1145/1015706.1015745

M K Misztal, K Erleben, A Bargteil, J Fursund, B Bunch Christensen, J A Bærentzen, and R Bridson. 2012. Multiphase Flow of Immiscible Fluids on Unstructured Moving Meshes. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '12)*. Eurographics Association, Goslar Germany, Germany, 97–106. http://dl.acm.org/citation.cfm?id=2422356.2422372

Neil Molino, Zhaosheng Bao, and Ron Fedkiw. 2004. A Virtual Node Algorithm for Changing Mesh Topology during Simulation. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 385–392. https://doi.org/10.1145/1015706.1015734

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

Rahul Narain, Tobias Pfaff, and James F. O'Brien. 2013. Folding and crumpling adaptive sheets. *ACM Transactions on Graphics* 32, 4 (jul 2013), 1. https://doi.org/10.1145/2461912.2462010

Rahul Narain, Armin Samii, and James F. O'Brien. 2012. Adaptive anisotropic remeshing for cloth simulation. *ACM Transactions on Graphics* 31, 6 (nov 2012), 1. https://doi.org/10.1145/2366145.2366171

Matthieu Nesme, Paul G. Kry, Lenka Jeřábková, and François Faure. 2009. Preserving topology and elasticity for embedded deformable models. In *ACM SIGGRAPH 2009 papers on - SIGGRAPH '09*. ACM Press, New York, New York, USA, 1. https://doi.org/10.1145/1576246.1531358

Yen Ting Ng, Chohong Min, and Frédéric Gibou. 2009. An Efficient Fluid-solid Coupling Algorithm for Single-phase Flows. *J. Comput. Phys.* 228, 23 (2009), 8807–8829. https://doi.org/10.1016/j.jcp.2009.08.032

N. Rasmussen, D. Enright, D. Nguyen, S. Marino, N. Sumner, W. Geiger, S. Hoon, and R. Fedkiw. 2004. Directable photorealistic liquids. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '04*. ACM Press, New York, New York, USA, 193. https://doi.org/10.1145/1028523.1028549

Peter Schwartz, Michael Barad, Phillip Colella, and Terry Ligocki. 2006. A Cartesian grid embedded boundary method for the heat equation and Poisson's equation in three dimensions. *J. Comput. Phys.* 211, 2 (2006), 531 – 550. https://doi.org/10.1016/j.jcp.2005.06.010

Rajsekhar Setaluri, Mridul Aanjaneya, Sean Bauer, and Eftychios Sifakis. 2014. SPGrid. *ACM Transactions on Graphics* 33, 6 (nov 2014), 1–12. https://doi.org/10.1145/2661229.2661269

Lin Shi and Yizhou Yu. 2004. Visual smoke simulation with adaptive octree refinement.

Eftychios Sifakis, Tamar Shinar, Geoffrey Irving, and Ronald Fedkiw. 2007. Hybrid Simulation of Deformable Solids. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '07)*. Eurographics Association, Goslar Germany, Germany, 81–90. http://dl.acm.org/citation.cfm?id=1272690.1272702

Alexey Stomakhin and Andrew Selle. 2017. Fluxed animated boundary method. *ACM Transactions on Graphics* 36, 4 (jul 2017), 1–8. https://doi.org/10.1145/3072959.3073597

Mark Sussman and Mitsuhiro Ohta. 2009. A Stable and Efficient Method for Treating Surface Tension in Incompressible Two-Phase Flow. *SIAM Journal on Scientific Computing* 31, 4 (jan 2009), 2447–2471. https://doi.org/10.1137/080732122

Tetsuya Takahashi and Ming C. Lin. 2019. A Geometrically Consistent Viscous Fluid Solver with Two-Way Fluid-Solid Coupling. *Computer Graphics Forum* 38, 2 (may 2019), 49–58. https://doi.org/10.1111/cgf.13618

Michael Tao, Christopher Batty, Eugene Fiume, and David I. W. Levin. 2019. Mandoline. *ACM Transactions on Graphics* 38, 6 (nov 2019), 1–17. https://doi.org/10.1145/3355089.3356543

J. Teran, E. Sifakis, S.S. Blemker, V. Ng-Thow-Hing, C. Lau, and R. Fedkiw. 2005. Creating and Simulating Skeletal Muscle from the Visible Human Data Set. *IEEE Transactions on Visualization and Computer Graphics* 11, 3 (may 2005), 317–328. https://doi.org/10.1109/TVCG.2005.42

Nils Thürey, Chris Wojtan, Markus Gross, and Greg Turk. 2010. A multiscale approach to mesh-based surface tension flows. *ACM Transactions on Graphics* 29, 4 (jul 2010), 1. https://doi.org/10.1145/1778765.1778785

Nils Thürey, Chris Wojtan, Markus Gross, and Greg Turk. 2010. A Multiscale Approach to Mesh-Based Surface Tension Flows. *ACM Trans. Graph.* 29, 4, Article Article 48 (July 2010), 10 pages. https://doi.org/10.1145/1778765.1778785

Huamin Wang, Peter J. Mucha, and Greg Turk. 2005. Water drops on surfaces. *ACM Transactions on Graphics* 24, 3 (jul 2005), 921. https://doi.org/10.1145/1073204.1073284

Martin Wicke, Daniel Ritchie, Bryan M. Klingner, Sebastian Burke, Jonathan R. Shewchuk, and James F. O'Brien. 2010. Dynamic local remeshing for elastoplastic simulation. *ACM Transactions on Graphics* 29, 4 (jul 2010), 1. https://doi.org/10.1145/1778765.1778786

Chris Wojtan, Nils Thürey, Markus Gross, and Greg Turk. 2009. Deforming meshes that split and merge. *ACM Transactions on Graphics* 28, 3 (jul 2009), 1. https://doi.org/10.1145/1531326.1531382

Chris Wojtan, Nils Thürey, Markus Gross, and Greg Turk. 2010. Physics-inspired topology changes for thin fluid features. In *ACM SIGGRAPH 2010 papers on - SIGGRAPH '10*. ACM Press, New York, New York, USA, 1. https://doi.org/10.1145/1833349.1778787

Chris Wojtan and Greg Turk. 2008. Fast viscoelastic behavior with thin features. In *ACM SIGGRAPH 2008 papers on - SIGGRAPH '08*. ACM Press, New York, New York, USA, 1. https://doi.org/10.1145/1399504.1360646

Omar Zarifi and Christopher Batty. 2017. A positive-definite cut-cell method for strong two-way coupling between fluids and deformable bodies. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation - SCA '17*. ACM Press, New York, New York, USA, 1–11. https://doi.org/10.1145/3099564.3099572

Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. 1997. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Software* 23, 4 (dec 1997), 550–560. https://doi.org/10.1145/279232.279236

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Transactions on Graphics* 24, 3 (jul 2005), 965. https://doi.org/10.1145/1073204.1073298