Codimensional Surface Tension Flow using Moving-Least-Squares Particles

HUI WANG, Shanghai Jiao Tong University
YONGXU JIN, Stanford University
ANQI LUO, Dartmouth College
XUBO YANG*, Shanghai Jiao Tong University
BO ZHU*, Dartmouth College

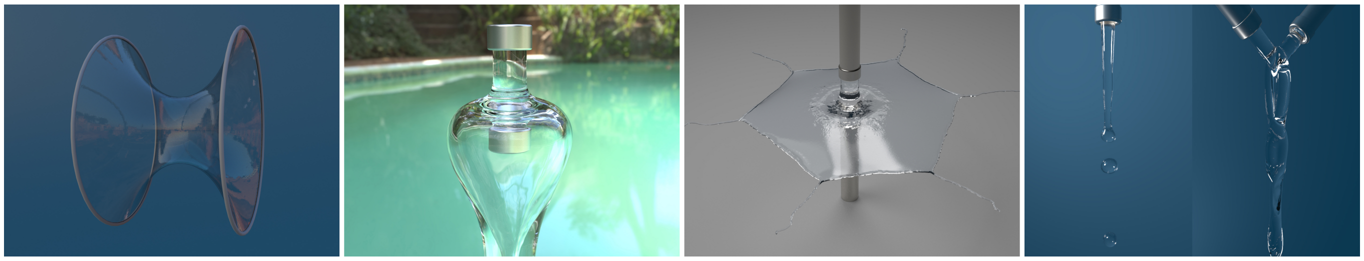


Fig. 1. Various codimensional fluid phenomena simulated using our codimensional MLS particle method. (Far Left) Film catenoid: A membrane attached to two rings contracts due to surface tension. (Middle Left) Water bell: Two vertical jets of water strike together, resulting in a bell-like water sheet. (Middle Right) Fluid polygon: Two vertical jets of water collide and form a polygonal thin sheet with pinched off filaments. (Far Right) Dripping water and fluid chain: Water flows from a pipe and forms droplets; Two jets collide and form a chain-like structure.

We propose a new Eulerian-Lagrangian approach to simulate the various surface tension phenomena characterized by volume, thin sheets, thin filaments, and points using Moving-Least-Squares (MLS) particles. At the center of our approach is a meshless Lagrangian description of the different types of codimensional geometries and their transitions using an MLS approximation. In particular, we differentiate the codimension-1 and codimension-2 geometries on Lagrangian MLS particles to precisely describe the evolution of thin sheets and filaments, and we discretize the codimension-0 operators on a background Cartesian grid for efficient volumetric processing. Physical forces including surface tension and pressure across different codimensions are coupled in a monolithic manner by solving one single linear system to evolve the surface-tension driven Navier-Stokes system in a complex non-manifold space. The codimensional transitions are handled explicitly by tracking a codimension number stored on each particle, which replaces the tedious meshing operators in a conventional mesh-based approach. Using the proposed framework, we simulate a broad array of visually appealing surface tension phenomena, including the fluid chain, bell, polygon, catenoid, and dripping, to demonstrate the efficacy of our approach in capturing the complex fluid characteristics with mixed codimensions, in a robust, versatile, and connectivity-free manner.

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

Additional Key Words and Phrases: codimensional fluids, surface tension, PIC/FLIP, moving-least-squares

ACM Reference Format:
Hui Wang, Yongxu Jin, Anqi Luo, Xubo Yang, and Bo Zhu. 2020. Codimensional Surface Tension Flow using Moving-Least-Squares Particles. *ACM Trans. Graph.* 39, 4, Article 42 (July 2020), 16 pages. https://doi.org/10.1145/3386569.3392487

1 INTRODUCTION

Particle methods are inherently adaptive. With particles, a broad array of interesting interfacial fluid phenomena can be modeled effectively by deploying a sufficient number of point samples around the region of interest to produce high-quality simulations. Typical examples can be seen in particle level-set methods [Enright et al. 2002], adaptive SPH [Adams et al. 2007; Winchenbach et al. 2017], adaptive MPM [Gao et al. 2017], narrow-band PIC/FLIP [Ferstl et al. 2016], etc. All these methods build the discrete differential operators in a volumetric way, either on particles directly or on an auxiliary background grid, by treating each particle as an isotropic point sample in space. The success of these computational paradigms relies on a well-defined sample distribution over the computational domain, which needs to be sufficiently dense and regular, to carry out local approximations by smoothed particle quantities using a mollified kernel function (see [Cottet et al. 2000]). However, on the minus side, particle methods are not inherently codimensional. It is extremely challenging to simulate complex fluid systems composed of volumes and codimensional features. These feature types include

*Corresponding authors

Authors' addresses: Hui Wang, wanghehv@sjtu.edu.cn, Shanghai Jiao Tong University; Yongxu Jin, yxjin@stanford.edu, Stanford University; Anqi Luo, anqi.luo.gr@dartmouth.edu, Dartmouth College; Xubo Yang, yangxubo@sjtu.edu.cn, Shanghai Jiao Tong University; Bo Zhu, bo.zhu@dartmouth.edu, Dartmouth College.

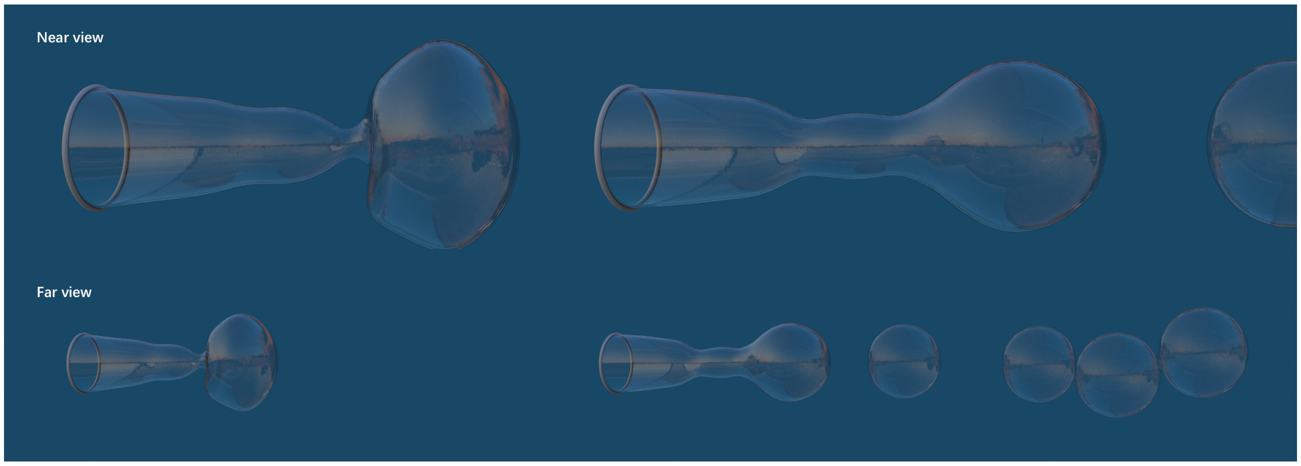


Fig. 2. Blowing bubble: The soap membrane is stretched due to the wind flow and generates individual bubbles.

fluid volume (codimension-0), thin sheets (codimension-1), filaments (codimension-2), and points (codimension-3), which are commonly seen in a variety of phenomena such as "fluid polygon" [Buckingham and Bush 2001] and "waterbell" [Clanet 2007].

Despite the pioneering research done in computer graphics on enabling particles to simulate physics on thin geometries (e.g., see [Ando et al. 2012; Martin et al. 2010]), the vast majority of previous literature on the numerical modeling of a codimensional system relies on the leverage of a mesh structure [Chang et al. 2019; Zhu et al. 2015, 2014]. Regardless of the places where a computational stencil is actually built, e.g. on particles [Zhu et al. 2014] or on a background grid [Jiang et al. 2017], the mesh connectivity information plays a pivotal role in describing the local geometry and accommodating the physics differentiation and domain evolution. On the computational side, the maintenance of a valid mesh structure is expensive, particularly in a highly dynamic physical setting, when the mesh topology needs to be updated frequently. In addition, the difficulties in implementing robust and high-performance meshing code that can automatically perform the various remeshing operations (e.g., [Brochu and Bridson 2009; Da et al. 2016; Weidner et al. 2018]) raise the bar for a mesh-based algorithm to be widely used.

In this paper, we tackle the computational dilemma between particles and mesh by introducing a mesh-free Lagrangian representation based on MLS particles to model a codimensional system characterized by complex geometric features and dynamic evolutions. The fundamental philosophy of our approach is to *"divide-and-couple"* particles with different types. Unlike a traditional meshfree method, which treats all features using one unified isotropic kernel, we subdivide the particles into different groups according to their codimensions and specify their transition, coupling, and evolution in an explicit fashion. In this sense, a particle system enhanced by its codimension information, which can be thought of as "weak" or "smoothed" connectivities, can resemble the functions of a codimensional mesh when describing the local geometric features without maintaining its mesh elements.

We designed two essential mechanisms in our algorithm to enable a particle system to accommodate the geometric evolution and numerical PDE discretization in a non-manifold setting. First, we assign a codimension number to each particle to distinguish its various numerical behaviors among the four different codimension groups. This codimension number is updated for each particle when a codimension transition event happens (e.g., when a volume particle gets merged to a surface). Second, we build three sets of differential operators, one for each codimension, to solve a numerical PDE discretized on the codimensional particle system. In particular, we devise the codimension-1 and codimension-2 operators based on an MLS method to approximate the local geometry on a point cloud [Liang et al. 2012; Liang and Zhao 2013]. The codimension-0 operators are discretized on a background grid for efficiency. The communications across different codimensional groups are restricted stringently by predefining all the feasible coupling effects, e.g., the differential operator coupling between thin sheets and filaments, in a physically-plausible way.

With these two numerical tools in hand, we are able to design an efficient and versatile approach to solve the codimensional Navier-Stokes equations on a meshfree, non-manifold discretization, with its efficacy demonstrated by a broad range of complex surface tension phenomena, including bubbles, film catenoids, dripping water, waterbells, and fluid polygons. Compared with previous literature, our method shows its unique merits in robustly and efficiently evolving a PDE-driven dynamic system characterized by various codimensional features without maintaining explicit connectivity.

The main contributions of our work can be summarized as:

- The first mesh-free Lagrangian solver to model complex codimensional fluids exhibiting rich thin features.
- A novel "divide-and-couple" mechanism to model the evolution and transitions of a codimensional particle system.
- A set of discrete operators based on MLS approximation to solve PDEs on a codimensional mesh-free discretization.
- A monolithic coupling algorithm to model the codimensional surface tension and incompressibility in one linear system.

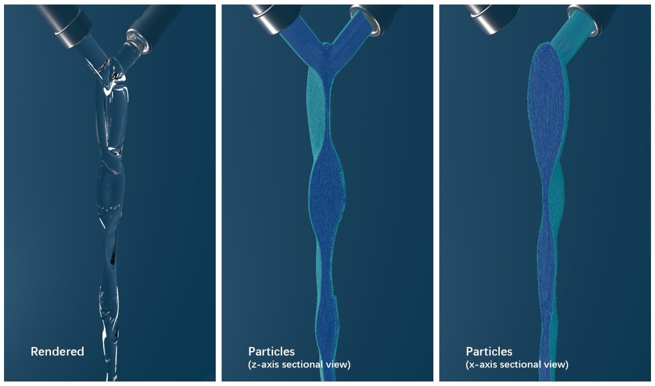


Fig. 3. Fluid chain: Two jets collide and form a chain-like structure due to the surface tension and the inertia.

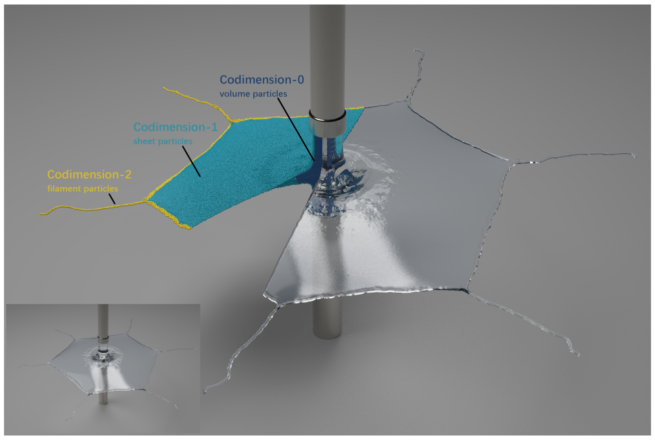


Fig. 4. Overview of our MLS particle method. The fluid volume is discretized to meshless particles with codimension numbers, and the physical forces across different codimensions are solved in a monolithic coupled way.

2 RELATED WORK

Mesh-based methods. On a Lagrangian mesh with connectivity information, a mesh surface can be used to accommodate the surface tension computation, with the differential operators defined on a mesh node and its incident triangles [Crane 2018; Zheng et al. 2015]. Zhang et al. [2011] simulate droplets on a surface mesh using deformation operators. Batty et al. [2012] model thin viscous sheets with the single-layer mesh with thickness. Da et al. [2015] model complex bubble structures with a Lagrangian non-manifold triangle mesh and integrate the surface tension into vertex-based circulations. Da et al. [2016] develop a surface-only simulation framework that reduces a general 3D fluid solver onto the surface mesh. Ishida et al. [2017] replace the curvature by the gradient of the surface area functional, to handle the non-manifold junctions of films and bubbles robustly. Zhu et al. [2015; 2014] propose a unified framework that simulates codimensional phenomena using codimensional simplicial complexes. Jiang et al. [2017] integrate the Lagrangian meshes into the MPM framework and propose a codimensional hybrid update of the deformation gradient to simulate elastoplastic materials like clothes and hair. Guo et al. [2018] further improve this method to support thin shells.

Particle-based and hybrid methods. In general, there are two computational paradigms of particle-based surface tension. An interparticle interaction force (IIF) method [Becker and Teschner 2007; Hong et al. 2008a; Tartakovsky and Meakin 2005] models the surface tension as a cohesive force between the interface and the interior particles, while a continuum surface force (CSF) method [He et al. 2014; Hu and Adams 2006; Müller et al. 2003] models the surface tension as an interfacial energy minimization on the surface area. By combining the IIF and CSF models, Akinci et al. [2013] propose a method to robustly model surface tension on SPH particles. Wang et al. [2017] incorporate this model into IISPH [Ihmsen et al. 2013] framework. We refer readers to [Huber et al. 2015] for a benchmark of different particle-based surface tension methods. As for particle-based viscosity, a series of implicit solvers for SPH fluids have been proposed to enhance the various viscous simulation effects [Peer and Teschner 2016; Takahashi et al. 2015; Weiler et al. 2018]. Hybrid particle-mesh methods have been drawing increasing attention recently, including PIC [Harlow 1962; Harlow and Welch 1965], FLIP [Brackbill et al. 1988; Brackbill and Ruppel 1986] and their combinations [Zhu and Bridson 2005]. The APIC [Jiang et al. 2015a] method addresses the dissipation issue of PIC by adding locally affine descriptions. For surface tension in hybrid methods, Boyd et al. [2012] construct a level set with the particle level set (PLS) method and compute surface tension on it, Thürey et al. [2010] simulate surface tension on the grid for stability and on the mesh for sub-grid detail, and Schroeder et al. [2012] apply semi-implicit surface tension on a Lagrangian mesh to a Eulerian discretization.

Point-based surface. Surface differential operators can be approximated by various methods. In smoothed particle hydrodynamics (SPH) [Gingold and Monaghan 1977; Monaghan 1992], the differential operators are approximated by radial symmetrical smoothing kernels. The traditional SPH method lacks the capability of conducting differential operations on a codimensional manifold; Petronetto et al. [2013] propose an SPH-style Laplacian-Beltrami operator [Schmidt 2014] with an optimization mechanism to estimate the area of the local element. Other approximation methods include graph Laplacian [Belkin and Niyogi 2008], local triangular mesh [Belkin et al. 2009; Lai et al. 2013], closest point method (CPM) [Cheung et al. 2015; Ruuth and Merriman 2008], and moving least squares (MLS) [Lancaster and Salkauskas 1981; Levin 1998; Nealen 2004]. CPM embeds the function defined on codimensional manifold to a Cartesian space and interpolates the result back to the manifold after solving PDE in the Cartesian space. It can be used for different underlying discretizations, such as particles, meshes, implicit surfaces, and different codimensional structures. Some works [Auer et al. 2012; Auer and Westermann 2013; Kim et al. 2013] adopted CPM to solve surface flow. Saye et al. [2014] present an efficient method for calculating high-order approximations of closest points on implicit surfaces. By tracking the particle-sampled surface using grid-based particle method (GBPM) [Leung and Zhao 2009a,b], Leung et al. [2011] solve PDEs on the evolving surface.

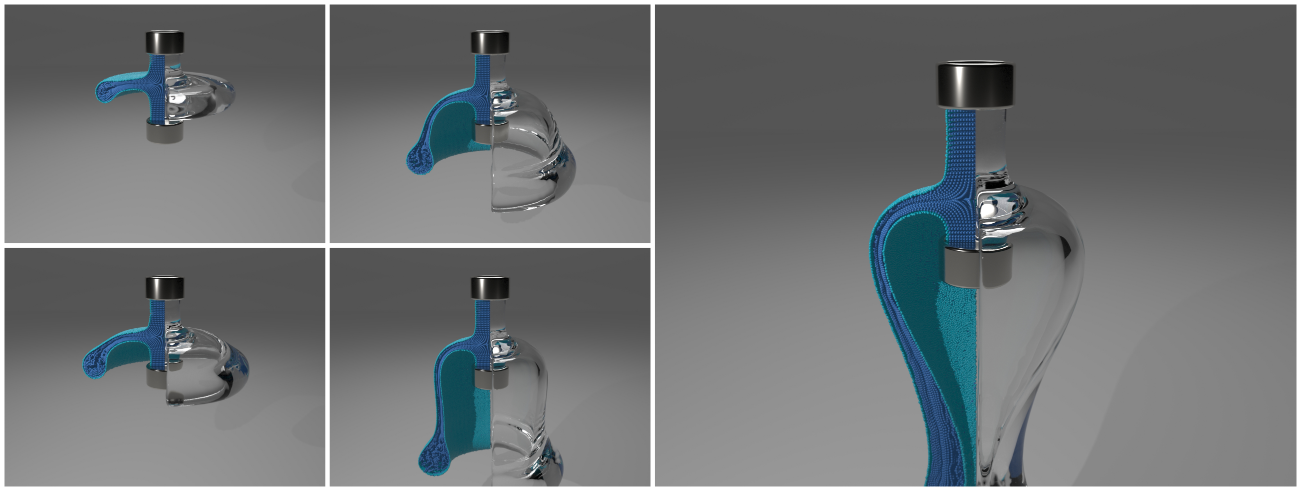


Fig. 5. Waterbell: Two vertical jets collide and generate a circular fluid sheet, resulting in a bell shape

Moving Least Squares. The Moving-Least-Squares method was proposed for continuous function approximation based on point cloud data. It can be used in surface approximation [Alexa et al. 2001; Levin 1998], surface rendering [Ledergerber et al. 2008] or PDE solving [Liang et al. 2012; Liang and Zhao 2013]. In particular, Alexa et al. [2001] reconstruct a smooth manifold from a set of points by creating local maps and approximating these local maps by the MLS method. The computer graphics community uses the MLS method to simulate elastic/plastic materials [Müller et al. 2004], fractures [Pauly et al. 2005], and fluids [Band et al. 2018, 2017; Hong et al. 2008b; Shobeyri and Afshar 2010]. Hu et al. [2018] integrate the MLS method into an MPM framework, leading to a unified implementation and a significant speed-up of the simulation pipeline. Zhang et al. [2010] add the surface tension onto SPH-based particles by approximating curvature using the MLS method.

3 CODIMENSIONAL GEOMETRY

Codimensional particles. We discretize the volume of fluid in a Lagrangian way by a set of meshless particles. Each particle carries a set of physical properties such as position $\vec{x}$, velocity $\vec{u}$, and mass m. In addition, each particle carries an integer codimension number C, specifying the codimensionality of its local geometry. Specifically, we have $C = 0$ for volume, $C = 1$ for thin films, $C = 2$ for thin filaments, and $C = 3$ for discrete points. We divide the particles into four groups according to their codimension numbers that are initialized properly at the beginning. For example, all the particles on the boundary have codimension number $C = 1$ and all the particles in the interior have codimension number $C = 0$. This number is updated during the simulation steps by tracking the codimension transition events. For example, we have $C : 0 \longrightarrow 1$, when a volumetric particle is merged into an interface, or $C : 1 \longrightarrow 2$, when a thin sheet particle is pinched off into a filament particle, as in Figure 4. The codimensional transition events are captured by monitoring the local particle distribution. We refer the readers to Section 8 for a more detailed discussion.

3.1 Operator definition

Naming convention. We use a superscript to indicate a coordinate index. For example, v^k stands for the k-th component of a vector field $\vec{v}$. We use a subscript to denote the particle index. For example, $\vec{x}_q$ stands for the position of the q-th particle. We also use subscripts to denote the index of the polynomial basis. We use a subscript within parentheses to denote the codimension for the operator, e.g., $\nabla^2_{(1)}$ is the Laplacian operator defined on codimension-1 particles.

Next, we define the three differential operators on a codimensional shape. Given a codimension-i geometry in $\mathbb{R}^d$, with $i \in [0, 2]$, we write the unified formulae for the gradient of a scalar field s, the divergence of a vector field $\vec{v}$, and the Laplacian of a scalar field s as

$$\begin{cases} \nabla_{(i)} s = \sum_{k=1}^{d-i} \left(\sum_{l=1}^{d-i} g^{kl} \frac{\partial s}{\partial \xi^l} \right) \vec{t}^k, & \text{(1a)} \\ \nabla_{(i)} \cdot \vec{v} = \frac{1}{\sqrt{g}} \sum_{k=1}^{d-i} \frac{\partial}{\partial \xi^k} (\sqrt{g} v^k), & \text{(1b)} \\ \nabla^2_{(i)} s = \frac{1}{\sqrt{g}} \sum_{k,l=1}^{d-i} \frac{\partial}{\partial \xi^k} (\sqrt{g} g^{kl} \frac{\partial s}{\partial \xi^l}), & \text{(1c)} \end{cases}$$

where $k, l \in [1, d-i]$ are the coordinate indices, $\vec{t}^k$ stands for the k-th basis of the local tangent space, ξ^k is the k-th local coordinate, and v^k is the k-th component of $\vec{v}$ that satisfies $\vec{v} = \sum_k v^k \vec{t}^k$. We define $[g_{kl}]$ as the $(d-i) \times (d-i)$ metric tensor, $[g^{kl}]$ as its inverse, and $g = \det([g_{kl}])$ as its determinant. We refer the readers to [Jost 2008] for a detailed description of the Riemannian manifolds.

To instantiate Equation (1) for a specific codimension, we first consider a volume $\mathcal{V} \subset \mathbb{R}^3$ $(i = 0)$ that is parameterized by three parameters (ξ^1, ξ^2, ξ^3) in a three-dimensional Cartesian coordinate system where the basis $\vec{t}^k$ is orthonormal. The metric tensor $[g_{kl}]$ becomes a 3×3 identity matrix and the codimension-0 differential

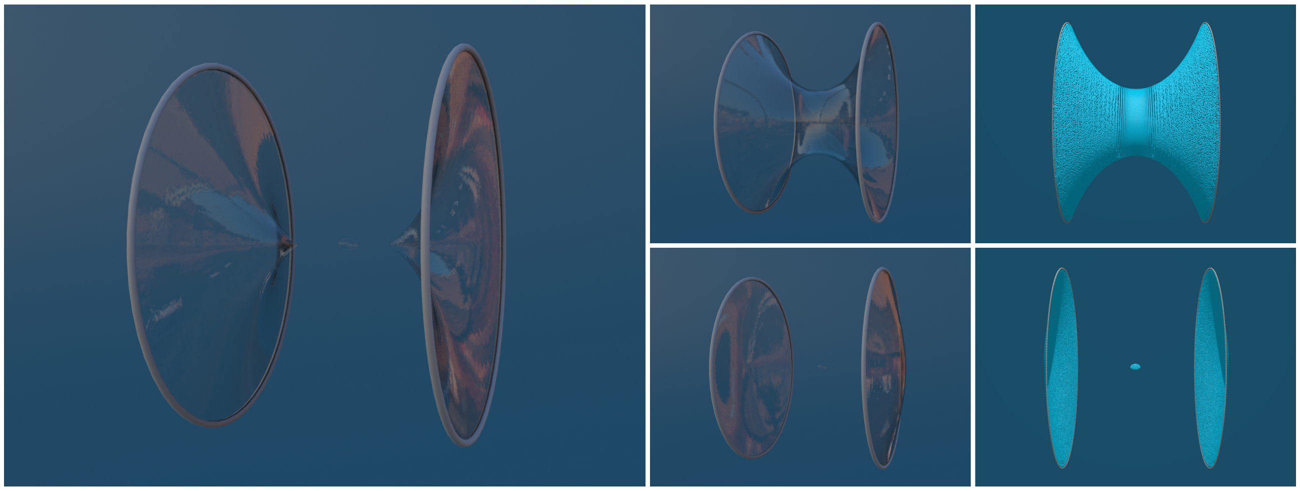


Fig. 6. Film catenoid: The thin membrane is attached to two rings. As two rings separate, the membrane contracts, splits into two surfaces and generates a bubble in the middle.

operators have simpler formulae as

$$\begin{cases} \nabla_{(0)} s = \sum_{k=1}^{3} \left(\frac{\partial s}{\partial \xi^k} \right) \vec{t}^k, & (2a) \\ \nabla_{(0)} \cdot \vec{v} = \sum_{k=1}^{3} \frac{\partial v^k}{\partial \xi^k}, & (2b) \\ \nabla^2_{(0)} s = \sum_{k=1}^{3} \frac{\partial^2 s}{(\partial \xi^k)^2}. & (2c) \end{cases}$$

For the case of codimension-1, we consider a surface $\mathcal{S} \subset \mathbb{R}^3$ ($i = 1$) with its local space parameterized by two parameters (ξ^1, ξ^2). Similarly, we have a filament $\mathcal{F} \subset \mathbb{R}^3$ ($i = 2$) that is parameterized by a single parameter ξ^1. The differential operators for these two codimensional cases can be defined by substituting the values of i and d into Equation (1).

3.2 Operator discretization

The three differential operators introduced above are discretized in two different manners according to the local codimension number C. In particular, for a volumetric operator ($C = 0$), we leverage a background grid to establish the Eulerian computational stencils by interpolating the quantities back-and-forth between the particles and the grid cells. For a thin-sheet or thin-filament operator ($C = 1, 2$), we employ a Moving-Least-Squares approximation to create the local basis and devise the differential stencils purely from the Lagrangian particles. We do not employ a differential operator on a discrete point ($C = 3$).

3.2.1 Codimension-0 operators. In our framework, codimension-0 operators are discretized in a Eulerian way to take advantage of its Cartesian nature. Following the convention in PIC/FLIP [Zhu and Bridson 2005], the discretized differential stencils are built on a background MAC grid by first transferring the data from particles to the grid, conducting the operator on the local grid cells and faces, and transferring the quantities back from the grid to particles. For the grid operation, the gradient, divergence, and Laplacian operators are discretized using a standard finite-difference stencil [Fedkiw et al. 2001] that is widely recognized in the computational physics and graphics communities.

3.2.2 Codimension-1 operators. We build the Lagrangian operators on a particle-sampled surface following the seminal work of [Liang and Zhao 2013] on solving point-cloud PDEs using the Moving-Least-Squares method. The key idea is to create a polynomial approximation for the local geometry around a particle and then build discrete operators on top of it. For a codimension-1 particle p, surrounded by n neighboring codimension-1 particles $\mathcal{N}_p = \{q \mid ||\vec{x}_p - \vec{x}_q|| < h\}$, with h as the local supporting radius, we define a local frame $\mathbf{F}_p$ with its origin co-locating with $\vec{x}_p$ and the three axes $\vec{b}^1, \vec{b}^2, \vec{b}^3$ aligned with the three eigenvectors of the local covariance matrix [Mitra and Nguyen 2003]. In particular, $\vec{b}^1$ and $\vec{b}^2$ are tangential to the local plane and $\vec{b}^3$ is the normal (see Appendix A for the detailed steps of building $\mathbf{F}_p$).

We use a polynomial $\hat{s}(\boldsymbol{\beta}, \xi^1, \xi^2)$ with two variables (ξ^1, ξ^2) and six coefficients $\boldsymbol{\beta} = [\beta_1, \beta_2, ..., \beta_6]^T$ to describe a local scalar field s as:

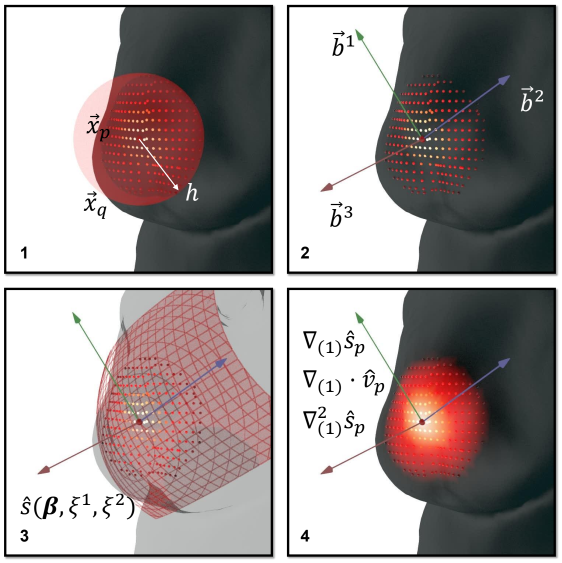
$$\hat{s}(\boldsymbol{\beta}, \xi^1, \xi^2) = \beta_1 + \beta_2 \xi^1 + \beta_3 \xi^2 + \beta_4 (\xi^1)^2 + \beta_5 (\xi^1 \xi^2) + \beta_6 (\xi^2)^2. \quad (3)$$

Given discrete values $\mathbf{s} = [s_1, s_2, ..., s_n]^T$ stored on particles, we can find the values of $\boldsymbol{\beta}$ by solving the following MLS problem:

$$\min_{\boldsymbol{\beta}} \sum_q \Psi(d_q)(\hat{s}(\boldsymbol{\beta}, \xi_q^1, \xi_q^2) - s_q)^2 \quad (4)$$

with $d_q = ||\vec{x}_p - \vec{x}_q||$ and Ψ is a local weighting function. Different $\boldsymbol{s}$ gives us different $\boldsymbol{\beta}$. In particular, if $\boldsymbol{s} = \boldsymbol{\xi}^3$, $\boldsymbol{\beta}$ encodes the local shape information that can be used to build the metric tensor $[g_{kl}]$.

We can write the MLS objective in Equation (4) in a matrix form as $(\mathbf{B}\boldsymbol{\beta} - \mathbf{s})^T \mathbf{W} (\mathbf{B}\boldsymbol{\beta} - \mathbf{s})$. $\mathbf{B}$ is an $n \times 6$ matrix with each row B_q as the polynomial basis vector of a particle q. The matrix $\mathbf{W}$ is an $n \times n$



$\vec{x}_p$
$\vec{x}_q$
h
1
$\vec{b}^1$
$\vec{b}^2$
$\vec{b}^3$
2
$\hat{s}(\boldsymbol{\beta}, \xi^1, \xi^2)$
3
$\nabla_{(1)}\hat{s}_p$
$\nabla_{(1)} \cdot \hat{v}_p$
$\nabla^2_{(1)}\hat{s}_p$
4

Fig. 7. Process of building differential operators on the surface using MLS method: (1) Find neighboring particles in supporting radius h; (2) Define the local frame $\mathbf{F}_p$; (3) Fitting the local surface and the local function (4) Build the codimensional operators

diagonal matrix with each diagonal term $\mathbf{W}_{qq}$ defined by $\Psi(d_q)$. Minimizing Equation (4) amounts to solving the normal equations

$$(\mathbf{B}^T\mathbf{W}\mathbf{B})\boldsymbol{\beta} = \mathbf{B}^T\mathbf{W}\mathbf{s}, \tag{5}$$

with the solution $\boldsymbol{\beta} = (\mathbf{B}^T\mathbf{W}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{W}\mathbf{s}$. It is noteworthy that $\mathbf{B}^T\mathbf{W}\mathbf{B}$ won't be singular because its diagonal terms are the weighted squared sum of the terms in the polynomial, which are always positive. By further defining $\mathbf{H} = (\mathbf{B}^T\mathbf{W}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{W}$, we can get a matrix expression for $\boldsymbol{\beta}$ as $\boldsymbol{\beta} = \mathbf{H}\mathbf{s}$.

When using $\mathbf{s} = \xi^3$, the second and the third components of $\boldsymbol{\beta}$ can be used to approximate the tangent vector basis as $\vec{t}^1 = (1, 0, \beta_2)$, $\vec{t}^2 = (0, 1, \beta_3)$ and the local metric tensor as

$$[g_{kl}] = \begin{bmatrix} \beta_2^2 + 1 & \beta_2\beta_3 \\ \beta_2\beta_3 & \beta_3^2 + 1 \end{bmatrix}. \tag{6}$$

With $\mathbf{H}$ and $[g_{kl}]$ in hand, we can define the codimension-1 differential operators given in Equation (1) on surface particles by substituting Equation (3) into Equation (1a)-(1c) with $i = 1$:

$$\begin{cases} \nabla_{(1)}s_p = (g^{11}\beta_2 + g^{12}\beta_3)\vec{t}^1 + (g^{21}\beta_2 + g^{22}\beta_3)\vec{t}^2, & \text{(7a)} \\ \nabla_{(1)} \cdot \vec{v}_p = \gamma_2 + \delta_3, & \text{(7b)} \\ \nabla^2_{(1)}s_p = 2g^{11}\beta_4 + (g^{12} + g^{21})\beta_5 + 2g^{22}\beta_6. & \text{(7c)} \end{cases}$$

Here we assume the two components of the vector field v_p^1 and v_p^2 are approximated by the polynomial coefficients $[\gamma_1, ..., \gamma_6]$ and $[\delta_1, ..., \delta_6]$ respectively. Similar to [Liang and Zhao 2013], we ignore the partial derivative of the metric tensor.

3.2.3 *Codimension-2 operators.*

We build the Lagrangian differential operators on filaments the same way as we did for thin sheets. We parameterize a filament using one parameter ξ^1, and run MLS to fit two quadratics to approximate the tangent vector basis and the metric tensor for the codimension-2 particles on the filament. In particular, we use the polynomials $\hat{s}(\boldsymbol{\beta}, \xi^1) = \beta_1 + \beta_2\xi^1 + \beta_3(\xi^1)^2$ to fit a scalar field and $\hat{v}^1(\boldsymbol{\gamma}, \xi^1) = \gamma_1 + \gamma_2\xi^1 + \gamma_3(\xi^1)^2$ to fit a vector field on a filament. The three codimension-2 operators can be defined as:

$$\begin{cases} \nabla_{(2)}s_p = (g^{11}\beta_2)\vec{t}^1, & \text{(8a)} \\ \nabla_{(2)} \cdot \vec{v}_p = \gamma_2, & \text{(8b)} \\ \nabla^2_{(2)}s_p = 2g^{11}\beta_3. & \text{(8c)} \end{cases}$$

Algorithm 1: Temporal evolution for a single timestep

```
1: Particle advection
2: Update codimension number $C$
3: Reseed $C$-1 and $C$-2 particles
4: Add body force on all particles
5: Interpolate velocities from $C$-0 particles to the grid
6: Solve viscosity on the grid and on $C$-1 and $C$-2 particles
7: Solve surface tension and incompressibility in a monolithic
   system coupling $C$-1 and $C$-2 particles and the grid
8: Correct velocities on the grid and interpolate the values back
   to $C$-0 particles using PIC/FLIP
```

3.2.4 *Codimension-1 and codimension-2 joint operators.*

We define the joint operator across codimension-1 and codimension-2 based on a geometric intuition by simply merging the two operators in one single expression, namely, by considering the thin sheet particles and the thin filament particles simultaneously in one operator. Thus, the size of the matrix $\mathbf{H}$ on a codimension-1 particle is extended to $6 \times (n_{(1)} + n_{(2)})$, and $3 \times (n_{(1)} + n_{(2)})$ for a codimension-2 particle, by incorporating the particles from both codimensions. For a better MLS approximation, we lower the local weights from neighboring particles with different codimensions. For example, we set the weight of codimension-1 particles to be 0.25 in a codimension-2 MLS fitting procedure for diffusion.

4 TIME INTEGRATION

With the codimensional Lagrangian operators in hand, we establish our numerical solver to model the evolution of a surface-tension-driven fluid system. For the physical model, we consider the Lagrangian form of the incompressible Navier-Stokes equations

$$\begin{cases} \rho\frac{D\vec{u}}{Dt} = -\nabla p + \mu\nabla^2\vec{u} + \vec{f}, & \text{(9a)} \\ \nabla \cdot \vec{u} = 0, & \text{(9b)} \end{cases}$$

where $\vec{u}$ is the velocity, ρ is the density, p is the pressure, μ is the viscosity coefficient, $\vec{f}$ represents external forces including gravity and surface tension.

Our temporal evolution scheme with codimensional particles starts from a standard PIC/FLIP solver [Zhu and Bridson 2005]. As shown in Algorithm 1, a standard PIC/FLIP solver follows the steps of advection, body force, particle-to-grid interpolation, viscosity, projection, and grid-to-particle interpolation (steps with black texts

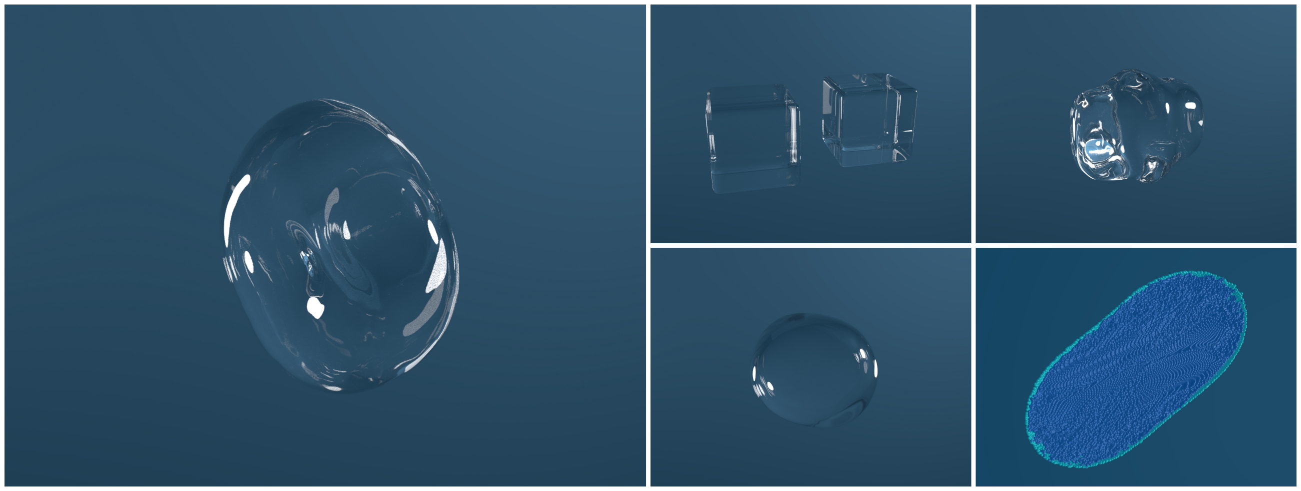


Fig. 8. Droplet collision: two droplets collide, merge and oscillate due to surface tension.

in Algorithm 1). Our codimensional particle method takes four additional steps (with colored texts in Algorithm 1) to enhance its ability in capturing codimensional features, including the codimension number update, codimensional particle reseeding, an MLS solve for codimensional viscosity, and a monolithic solve for the implicit coupling between surface tension and incompressibility. In particular, in the codimension number update step, we capture all the codimensional transition events and update their codimension number C accordingly. The particle reseeding step adds and deletes codimensional particles to maintain a uniform density distribution in the corresponding codimensional space. The viscosity step solves viscous forces on $C-1$ and $C-2$ particles in an implicit way using the MLS operators. The monolithic solve step couples the semi-implicit surface tension on $C-1$ and $C-2$ particles with the volumetric incompressibility condition on the background grid by setting impulse-based constraints, leading to a monolithic linear system which to solve the effects of surface tension and incompressibility simultaneously.

Our time integration scheme follows a splitting scheme [Fedkiw et al. 2001] by accumulating the effects from each force term in the Navier-Stokes equations onto intermediate velocities. In particular, by starting with $\vec{u}^n$, we get $\vec{u}^*$ after advection and applying gravity. It becomes $\vec{u}^{**}$ by solving viscosity and becomes $\vec{u}^{n+1}$ after the monolithic solve step.

5 VISCOSITY

To allow a large timestep and to reuse our codimensional Poisson solver, we take an implicit scheme to solve the viscosity in a component decoupled way (see [Rasmussen et al. 2004] for details). We want to point out that one of the main limitations of such decoupled scheme is that it enforces an inaccurate viscous boundary condition which could result in large damping effects near the interface (see [Batty and Bridson 2008]). An accurate, component-coupled scheme is a better option, especially for viscous flow simulations (e.g., see [Larionov et al. 2017]). We solve the volumetric viscosity on the background grid and the coupled sheet-and-filament viscosity on codimension-1 and codimension-2 particles. The two solvers follow the same governing equations (see below) with the subscript (i) denoting the codimension.

The viscosity term in Equation (9) can be split as

$$\rho \frac{D\vec{u}}{Dt} = \mu \nabla^2_{(i)} \vec{u}, \quad i = 0, 1, 2. \tag{10}$$

By defining particle volume $V = m/\rho$, we can rewrite Equation (10) for each particle as

$$m \frac{D\vec{u}}{Dt} = \mu V \nabla^2_{(i)} \vec{u}, \quad i = 0, 1, 2. \tag{11}$$

By taking an implicit time integration, we get

$$m\vec{u}^{**} = m\vec{u}^{*} + \mu V \nabla^2_{(i)} \vec{u}^{**} \Delta t, \quad i = 0, 1, 2, \tag{12}$$

which can be further written in a matrix form as

$$(\mathbf{M}_{(i)} - \mu \Delta t \mathbf{V}_{(i)} \mathbf{L}_{(i)}) \mathbf{u}^{**}_{(i)} = \mathbf{M}_{(i)} \mathbf{u}^{*}_{(i)}, \quad i = 0, 1, 2 \tag{13}$$

where $\mathbf{M}_{(i)}$ and $\mathbf{V}_{(i)}$ stand for the mass and volume matrix of the codimension-i group, and $\mathbf{L}_{(i)}$ stands for the codimension-i Laplacian matrix.

To solve viscosity on codimension-1 and codimension-2 particles together, we define the joint Laplacian matrix based on Section 3.2.4. For codimension-1 particles, the matrix is $[\mathbf{L}_{(1)} \mathbf{L}_{(1)(2)}]$ with $\mathbf{L}_{(1)(2)}$ denoting the contribution from codimension-2 particles. And for codimension-2 particles, the matrix is defined as $[\mathbf{L}_{(2)} \mathbf{L}_{(2)(1)}]$. Alternatively, an explicit viscosity solver can be implemented by replacing the $\vec{u}^{**}$ on RHS with $\vec{u}^{*}$ in Equation (12), at the expense of smaller timesteps.

6 SURFACE TENSION

We solve the semi-implicit surface tension force applied on a codimensional particle by leveraging the codimension-1 and codimension-2 differential operators derived in Section 3 to construct the linear system. This semi-implicit step will be coupled with the projection step to formulate a set of monolithic equations that will be solved

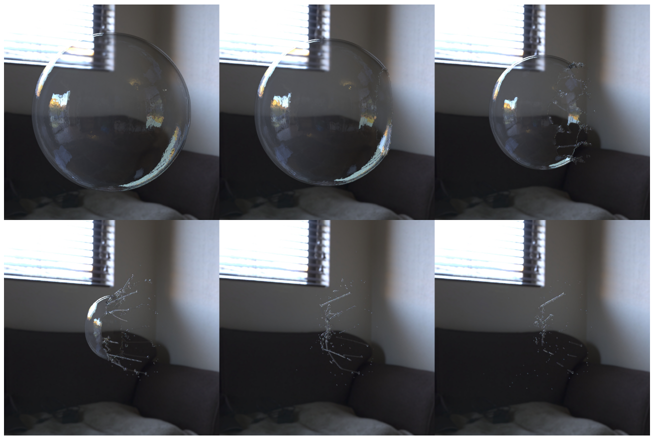


Fig. 9. Bursting bubble: The bubble is initialized with a hole. The rim of the hole shrinks inwards and pinches off thin filaments and points.

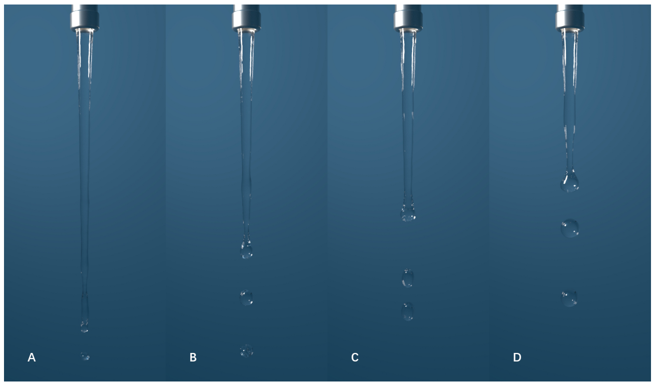


Fig. 10. Dripping water: The water drips from a pipe, forming the neck due to the surface tension. The droplets pinch off at the end of the stream afterward. The surface tension coefficients are 0.015, 0.03, 0.045 and 0.06 N/m from A to D.

for the velocity $\vec{u}^{n+1}$ for the next time step. In this section, we introduce the numerical discretization for the surface tension model on different codimensional geometries and we will introduce its implicit coupling with the pressure force in the next section.

Following the codimensional surface tension model proposed in [Zhu et al. 2014], we consider the surface tension effects on four different types of codimensional geometries – thin sheets, thin filaments, the rim (sheet boundary) and the bud (filament boundary). The surface tension force on a thin-sheet or thin-filament particle is defined as

$$\vec{f}_s = \Delta p S_{(i)} = \sigma\kappa\vec{n}S_{(i)} = \sigma\nabla^2_{(i)}\vec{x}S_{(i)}, \quad i = 1, 2, \tag{14}$$

where Δp is the pressure jump caused by surface tension on the fluid surface, $S_{(i)}$ is the control surface area (see Appendix D for a detailed discussion), σ is surface tension coefficient and $\kappa\vec{n}$ is the curvature vector calculated as $\kappa\vec{n} = \nabla^2_{(i)}\vec{x}$. Following the semi-implicit scheme for surface tension in [Zheng et al. 2006], we derive our semi-implicit codimensional surface tension time integration as

$$\begin{aligned} m\vec{u}^{n+1} &= m\vec{u}^{**} + \vec{f}_s^{n+1}\Delta t \\ &= m\vec{u}^{**} + \sigma\nabla^2_{(i)}\vec{x}^{n+1}S_{(i)}\Delta t \\ &= m\vec{u}^{**} + \sigma\nabla^2_{(i)}(\vec{x}^n + \vec{u}^{n+1}\Delta t)S_{(i)}\Delta t, \quad i = 1, 2, \end{aligned} \tag{15}$$

which can be solved as a linear system

$$(\mathbf{M}_{(i)} - \sigma\Delta t^2\mathbf{S}_{(i)}\mathbf{L}_{(i)})\mathbf{u}^{n+1}_{(i)} = \sigma\Delta t\mathbf{S}_{(i)}\mathbf{L}_{(i)}\mathbf{x}^n_{(i)} + \mathbf{M}_{(i)}\mathbf{u}^{**}_{(i)}, \quad i = 1, 2. \tag{16}$$

By further denoting $\bar{\mathbf{f}}_{(i)} = \sigma\mathbf{S}_{(i)}\mathbf{L}_{(i)}\mathbf{x}^n$ as the explicit part of the surface tension force and letting $\bar{\mathbf{L}}_{(i)} = \sigma\Delta t^2\mathbf{S}_{(i)}\mathbf{L}_{(i)}$, Equation (16) is simplified as

$$(\mathbf{M}_{(i)} - \bar{\mathbf{L}}_{(i)})\mathbf{u}^{n+1}_{(i)} = \Delta t\bar{\mathbf{f}}_{(i)} + \mathbf{M}_{(i)}\mathbf{u}^{**}_{(i)}, \quad i = 1, 2, \tag{17}$$

which will be integrated into the coupled system and solved together with the pressure (see details in Section 7).

Rim and bud surface tension. The rim surface tension force [Bush and Hasha 2004] is applied to those codimension-2 particles on the rim of the thin sheets, which causes the shrinkage of the sheet. Because the current MLS scheme cannot approximate a thickened rim geometry with a single layer of particles on the boundary of a thin sheet, we model the rim surface tension as an explicit force term which will be added on the right-hand side of the linear system. We approximate its direction $\vec{n}$ with $\vec{b}^2$ in $\mathbf{F}_p$, which is also the normal of the rim in the tangent plane. The rim curvature is the reciprocal of the particle radius r_p. According to Equation (14), we calculate the rim surface tension force as $\sigma\vec{n}S_{r,p}/r_p$ (see Appendix D for surface area approximation). We also model the explicit bud surface tension on those codimension-2 particles at the bud of each filament to deliver tangential shrinkage. The local geometry at the bud is treated numerically as a sphere with its radius as r_p.

7 CODIMESIONAL COUPLING

We model the coupling between the codimension-0 DoFs on a MAC grid and codimension-1 and codimension-2 DoFs on particles by introducing a new set of DoFs $\boldsymbol{\lambda}$ to model the impulse transferred between particles and the grid. Our coupling scheme modifies the monolithic two-way solid-fluid coupling algorithm proposed in [Robinson-Mosher et al. 2011] and [Schroeder et al. 2012] by introducing two sets of Lagrangian DoFs for codimension-1 and codimension-2 particles. The impulse is applied to the interface between the volume and the surface, which is discretized by a set of constraint faces on the MAC grid. We mark a cell that contains at least one codimension-0 particle as a fluid cell. And we mark a face that has at least one codimension-1 particle within its control volume as a constraint face. An equality constraint is enforced on every constraint face to guarantee that, for time step $n+1$, the codimension-0 velocity $\mathbf{v}^{n+1}_{(0)}$ stored on the faces equals the velocity interpolated from codimension-1 particles $\mathbf{u}^{n+1}_{(1)}$, which can be summarized in a matrix form as

$$\mathbf{K}\mathbf{v}^{n+1}_{(0)} = \mathbf{K}\mathbf{J}\mathbf{u}^{n+1}_{(1)} \tag{18}$$

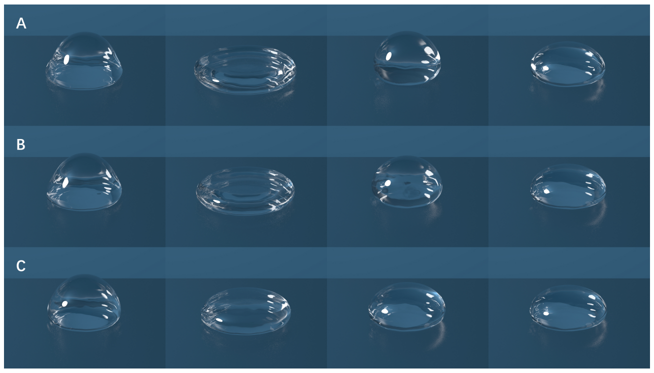


Fig. 11. Droplet on the plane: Droplets with different viscosities fall on the plane. The shape changes due to the impact of gravity and surface tension. We demonstrated the effects of different parameter sets. In particular, the viscosity for A to C are 0.01, 0.05, and 0.25 mPa·s.

with $\mathbf{K}$ as a selection matrix to pick out the constraint faces from all grid faces and $\mathbf{J}$ as an interpolation matrix that interpolates the codimension-1 particle velocities onto grid faces.

Next, we present four equations to describe the codimensional coupling between particles and the background grid. First, we give the pressure projection equation on the MAC grid by considering $\boldsymbol{\lambda}$ on the right-hand side as

$$\mathbf{M}_{(0)}\mathbf{v}_{(0)}^{n+1} - \mathbf{M}_{(0)}\mathbf{v}_{(0)}^{**} = -\Delta t \mathbf{G}_{(0)}\mathbf{p} + \mathbf{K}^T\boldsymbol{\lambda} \tag{19}$$

where $\mathbf{M}_{(0)}$ is the diagonal mass matrix for MAC faces, $\mathbf{G}_{(0)}$ is the volume-weighted gradient matrix on MAC faces and $\mathbf{p}$ is the pressure field. By defining $\bar{\mathbf{p}} = \Delta t\mathbf{p}$, and substituting Equation (19) into the divergence-free constraint $-\mathbf{G}_{(0)}^T\mathbf{v}_{(0)}^{n+1} = 0$, we have the Poisson equation with both the pressure $\bar{\mathbf{p}}$ and the impulse $\boldsymbol{\lambda}$ as unknowns:

$$\mathbf{G}_{(0)}^T\mathbf{M}_{(0)}^{-1}\mathbf{G}_{(0)}\bar{\mathbf{p}} - \mathbf{G}_{(0)}^T\mathbf{M}_{(0)}^{-1}\mathbf{K}^T\boldsymbol{\lambda} = \mathbf{G}_{(0)}^T\mathbf{v}_{(0)}^{**}. \tag{20}$$

Second, for the codimension-1 particles on a liquid surface, we write the momentum equation for semi-implicit surface tension by considering the contributions of both the impulse (from volume) and the Laplacian operator (from filaments) as

$$(\mathbf{M}_{(1)} - \bar{\mathbf{L}}_{(1)})\mathbf{u}_{(1)}^{n+1} - \bar{\mathbf{L}}_{(1)(2)}\mathbf{u}_{(2)}^{n+1} + \mathbf{J}^T\mathbf{K}^T\boldsymbol{\lambda} = \Delta t\bar{\mathbf{f}}_{(1)} + \mathbf{M}_{(1)}\mathbf{u}_{(1)}^{**} \tag{21}$$

where $\mathbf{J}^T\mathbf{K}^T$ maps the impulse to the surface particles. As we did for viscosity (see Section 5), here we use the symbol $\bar{\mathbf{L}}_{(1)(2)}$ to denote the Laplacian contribution from codimension-2 particles to codimension-1 particles.

Similarly, for the codimension-2 particles on filaments, we can write its momentum equation by coupling the codimensional interaction as

$$(\mathbf{M}_{(2)} - \bar{\mathbf{L}}_{(2)})\mathbf{u}_{(2)}^{n+1} - \bar{\mathbf{L}}_{(2)(1)}\mathbf{u}_{(1)}^{n+1} = \Delta t\bar{\mathbf{f}}_{(2)} + \mathbf{M}_{(2)}\mathbf{u}_{(2)}^{**}, \tag{22}$$

with the symbol $\bar{\mathbf{L}}_{(2)(1)}$ denoting the Laplacian contribution from codimension-1 particles to codimension-2 particles. Note that we did not consider the impulse from volume to filament in Equation (22).

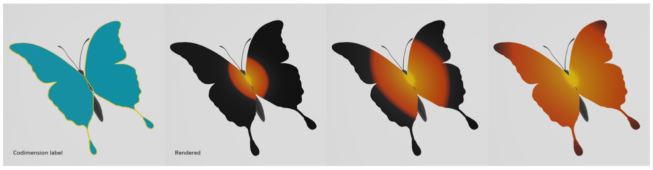


Fig. 12. Diffusion on butterfly: The butterfly wing is sampled by both surface (blue) and filament (yellow) particles. Our codimensional operator can solve the diffusion in the mixed-codimension scene.

Last, for the velocity constraints on grid faces, we substitute Equation (19) into Equation (18) to get:

$$-\mathbf{K}\mathbf{M}_{(0)}^{-1}\mathbf{G}_{(0)}\bar{\mathbf{p}} - \mathbf{K}\mathbf{J}\mathbf{u}_{(1)}^{n+1} + \mathbf{K}\mathbf{M}_{(0)}^{-1}\mathbf{K}^T\boldsymbol{\lambda} = -\mathbf{K}\mathbf{v}_{(0)}^{**}. \tag{23}$$

Combing Equation (20) (21) (22) and (23) yields a monolithically coupled linear system for codimension-0 grid pressure, codimension-1 particle velocity, codimension-2 particle velocity, and impulse:

$$\begin{bmatrix} \mathbf{G}_{(0)}^T\mathbf{M}_{(0)}^{-1}\mathbf{G}_{(0)} & 0 & 0 & -\mathbf{G}_{(0)}^T\mathbf{M}_{(0)}^{-1}\mathbf{K}^T \\ 0 & \bar{\mathbf{L}}_{(1)} - \mathbf{M}_{(1)} & \bar{\mathbf{L}}_{(1)(2)} & -\mathbf{J}^T\mathbf{K}^T \\ 0 & \bar{\mathbf{L}}_{(2)(1)} & \bar{\mathbf{L}}_{(2)} - \mathbf{M}_{(2)} & 0 \\ -\mathbf{K}\mathbf{M}_{(0)}^{-1}\mathbf{G}_{(0)} & -\mathbf{K}\mathbf{J} & 0 & \mathbf{K}\mathbf{M}_{(0)}^{-1}\mathbf{K}^T \end{bmatrix} \begin{bmatrix} \bar{\mathbf{p}} \\ \mathbf{u}_{(1)}^{n+1} \\ \mathbf{u}_{(2)}^{n+1} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{G}_{(0)}^T\mathbf{v}_{(0)}^{**} \\ -\Delta t\bar{\mathbf{f}}_{(1)} - \mathbf{M}_{(1)}\mathbf{u}_{(1)}^{**} \\ -\Delta t\bar{\mathbf{f}}_{(2)} - \mathbf{M}_{(2)}\mathbf{u}_{(2)}^{**} \\ -\mathbf{K}\mathbf{v}_{(0)}^{**} \end{bmatrix}. \tag{24}$$

It is noteworthy that in Equation (24), the rim surface tension and the bud surface tension are integrated into the system explicitly by adding additional force terms into the explicit part of the surface tension force $\bar{\mathbf{f}}_{(2)}$. Because the rim surface tension has already considered the contribution from the surface particles to the rim particles, we exclude the effects from the surface particles on the semi-implicit surface tension on rim particles by letting local weights of codimension-1 particles in codimension-2 MLS fitting be zeros. Due to the non-symmetric nature of the MLS Laplacian operator, we solve Equation (24) using the BiCGSTAB method [Van der Vorst 1992] with an incomplete-LU preconditioner and the tolerance 10^{-9}.

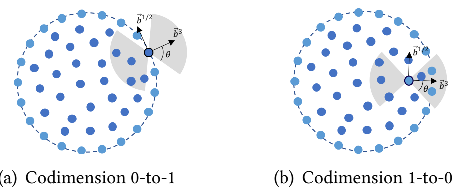
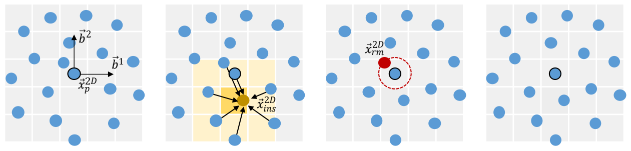


Fig. 13. Transition event detection: We use a pair of spherical cones to detect codimensional transition events.



(a) Project onto the tangent plane (b) Insert particle (c) Remove particle (d) Result

Fig. 14. Reseeding process: For each surface particle, the neighboring particles are projected onto its tangent space. The particle insertion and removal are then determined by the particle distribution on a 5×5 2D grid.

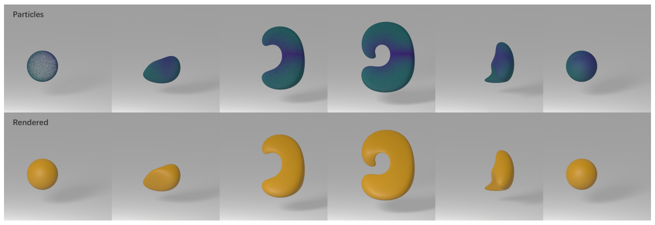


Fig. 15. Sphere deformation under a flow field: The initial sphere is sampled with particles sparsely and advected in a flow field. Our reseeding method can maintain a smooth particle-sampled surface even under great deformation.

8 CODIMENSIONAL TRANSITION

To ensure the codimensional compatibility, we enable the various codimensional transitions to ensure a consistent codimension number governed by the following rules:

Codimension 0-to-1 transition: A 0-to-1 transition happens when a volume particle is merged to a thin sheet, under either one of the two following conditions: (1) the particle is on one side of the codimension-1 interface while all the neighboring particles are on the other side; (2) the volume becomes sufficiently thin. To detect these conditions, we construct two opposite spherical cones with cone angle $\theta = \pi/3$ and radius h at the particle along the normal direction ($\vec{b}^3$) of the local frame $\mathbf{F}_p$. Condition (1) is satisfied if there is one and only one cone that contains codimension-0 or codimension-1 particles, as shown in Figure 13(a). Condition (2) is satisfied if both cones contain at least one codimension-1 particle and the minimum distance between the two sets of particles is below a threshold.

Codimension 1-to-0 transition: We employ the same strategy to detect the 1-to-0 transition. If both cones with cone angle $\theta = \pi/4$ along $\vec{b}^3$ contain at least one codimension-0 particle, the codimension-1 particle needs to be merged into the volume (see Figure 13(b)).

Codimension 1-to-2 transition: We employ a pair of cones with angle $\pi/6$, but aligned with $\vec{b}^2$ in $\mathbf{F}_p$, to detect the 1-to-2 transition. The transition will happen if one and only one cone contains codimension-1 or codimension-2 particles. We did not implement the codimension 2-to-1 mechanism although we could.

Codimension 2-to-3 transition: A 2-to-3 transition happens when there is no codimension-2 particle in its supporting radius. The continuity of the particles' velocity avoids the unnecessary reclassification of particles into different codimensional groups.

9 PARTICLE RESEEDING

The fluid may be stretched or compressed during the evolution, which will cause the change of surface area or filament length, necessitating a particle reseeding process (see [Jones et al. 2014] [Yue et al. 2015] [Gao et al. 2017] [Jiang et al. 2015b] [Ando et al. 2012] for examples). For a codimensional particle system, we reseed particles for different codimensions in separate passes. As shown in Figure 14, for a codimension-1 reseeding, we reseed new particles in a local frame $\mathbf{F}_p$ by following the steps of particle projection, insertion, and removal. First, we project the surrounding particles onto a 5×5 two-dimensional grid within the local tangential plane formed by $\vec{b}^1$ and $\vec{b}^2$ centered at $\vec{x}_p$. The 2D cell size is $0.4h$. Next, we iterate over the inner 3×3 cells and fill in a new particle for each empty cell that is fully surrounded by filled cells. The position of the new particle is then projected onto the local MLS surface. The mass and momentum are conserved during this process by redistributing the quantities from the existing particles to the new particles. Last, in the removal step, for every particle, we delete its closest neighbor if its distance is below a threshold between $0.1h$ and $0.2h$. For codimension-2 reseeding, we project particles onto a one-dimensional grid and conduct a similar process to the codimension-1 case. We did not employ a codimension-0 reseeding because our initial particle distribution and the smooth nature of surface tension flow avoid the significant particle distortion over the simulations. Adaptive schemes, such as [Adams et al. 2007; Ferstl et al. 2016; Winchenbach et al. 2017], can be considered for a better sampling strategy.

10 RESULTS

10.1 Numerical validation

Sphere diffusion. We compare the accuracy of our MLS-based PDE solver with other approaches including the closest point method [Ruuth and Merriman 2008] and the projected SPH method [Petronetto et al. 2013]. First, we solve a diffusion PDE $\frac{\partial s}{\partial t} = \nabla^2_{(1)} s$ on a unit sphere with the initial condition specified within a spherical coordinate system $s(\theta, \phi, 0) = cos\phi$. According to [Liang and Zhao 2013], the analytical solution at time t is given by $s(\theta, \phi, t) = e^{-2t} cos\phi$. We discretize the problem on a unit sphere sampled by 30k particles and solve it using the forward Euler method with $\Delta t = 0.001$. For comparison, we use the same kernel radius $h = 0.1$ in MLS and SPH and $\Delta x = 0.1$ as the cell size in CPM. Figure 17 shows the results of diffusion and the error. On a particle-sampled surface, CPM can only build the cell-particle mapping from the existing particles, which results in increasing error on cells close to the surface. The SPH-based method achieves a better accuracy(10^{-3}), while causing chaotic error distribution on the surface. The MLS method achieves the best accuracy(10^{-4}) and keeps a smooth error distribution on the surface.

Butterfly diffusion. To show the codimensional coupling ability of MLS on PDE solver, we solve a diffusion equation implicitly on

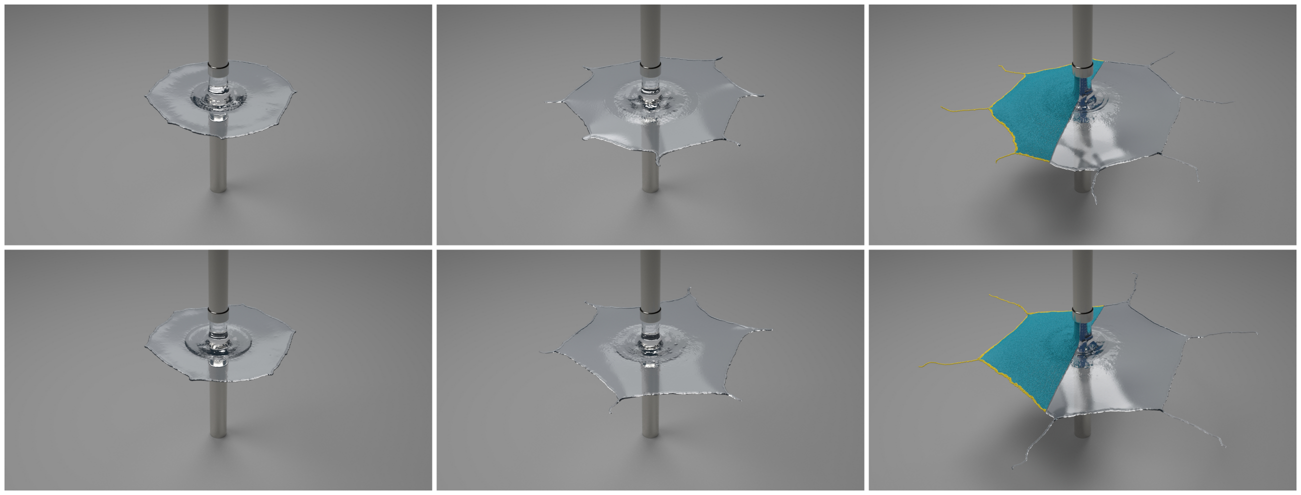


Fig. 16. Fluid polygon: Two jets collide vertically and generate a thin sheet. Multiplying the rim surface tension with a user-specified period function parameterized by the yaw angle of relative position produces the customized fluid polygon phenomena.

a butterfly-wing geometry, consisting of both codimension-1 and codimension-2 features. As shown in Figure 12, the rim of wings and thin tails are sampled by filament particles. The heat smoothly diffuses from the wing root to the entire region.

Sphere deformation. We demonstrate our reseeding method by showing a sphere deformation example in Figure 15. We advect the particles using the midpoint method with $\Delta t = 0.01s$ in a flow field proposed in [LeVeque 1996]. We start from a sparsely-sampled surface with about 50k particles. When the surface is stretched to the maximum at $t = 0.61s$, there are about 170k particles. Then the particle surface shrinks in an inverse flow field and restores its rest shape at $t = 1.22s$ with about 100k particles.

10.2 Examples

Here we show a number of simulations to showcase our codimensional particle solver. The physical parameters in the examples are listed in Appendix E. We rendered the liquid surface from particles directly using Houdini.

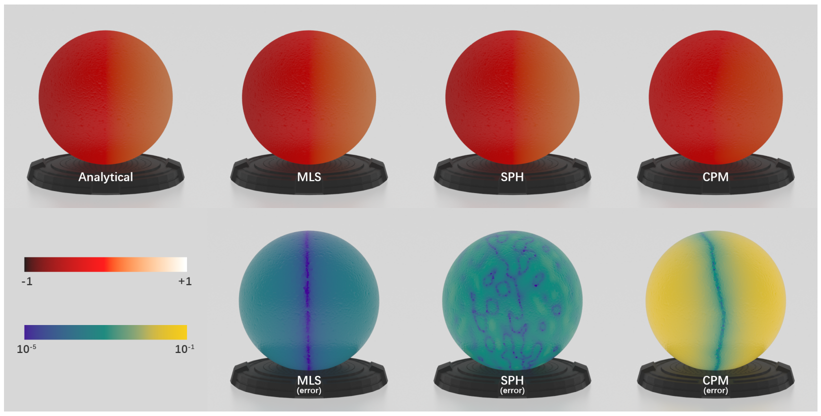
Blowing bubble, film catenoid, bursting bubble. Figure 2 shows that a membrane suspended on a ring expands because of the wind blowing and generates some separate bubbles. As shown in Figure 6, the thin membrane is attached to two rings. When two rings separate, the membrane contracts, separates into two membranes and generates a bubble in the middle. We simulate a bursting bubble in Figure 9. Due to the rim surface tension and the bud surface tension, the rim shrinks inwards and pinches off some filaments and droplets.

Droplet on the plane, droplet collision, dripping water. Figure 11 shows spherical droplets with various viscosities falling on a solid plane. The shape changes due to the impact of gravity and surface tension. In the droplet collision scene (see Figure 8), two cubic droplets with the opposite velocity collide and merge into one droplet. In the dripping water example, as shown in Figure 10, we show the water drips from a pipe with different surface tension coefficients. Due to the surface tension, the neck is formed and the droplets pinch off at the end of the stream afterward. This phenomenon is also called Rayleigh-Plateau instability.

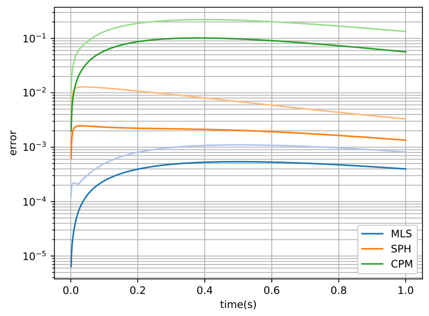
Fluid chain, waterbell, fluid polygon. In Figure 3, two opposite jets with the same velocity and radius collide with an angle of 90°. The fluid expands into a plate due to the inertia and converges due to the surface tension, forming the chain-like structure. The waterbell phenomenon [Clanet 2007] is implemented by two vertical jets colliding in Figure 5. As the generated plate falling, driven by surface tension, the lower part of the fluid converges and results in the bell-like shape. We simulate the fluid polygon mentioned in [Buckingham and Bush 2001]. To generate the user-specified fluid polygon, we multiply the rim surface tension with a period saw-tooth function parameterized by the yaw angle of the relative particle position. The result is shown in Figure 16.

10.3 Performance

We parallelize most of the steps in our method using OpenMP, and implement the GPU version of the BiCGSTAB solver using CUDA. Our examples are performed on a PC with an 8-core 3.40GHz CPU and NVIDIA GeForce GTX 1060 graphics card. The run time ranges widely depending on the complexity of the scene. The most expensive example is the waterbell example with about 14,000 active cells and 163,000 surface particles, which spends about 110s per frame including 35s for the viscosity step and 59s for the monolithic solve step. The bursting bubble example takes 22 seconds per frame with about 72k particles and 2 seconds when there are only 2k particles left. In the blowing bubble example, each individual bubble sampled with 20k-40k surface particles takes about 2-5 seconds to compute per frame. Besides, the average run time of the catenoid, droplet on the plane, droplet collision, dripping water, fluid chain, fluid polygon examples are 6s, 6.5s, 7s, 15s, 60s and 80s respectively.



(a) Diffusion result at 0.5s



(b) Error

Fig. 17. Heat diffusion on the sphere: We compare MLS with CPM[Ruuth and Merriman 2008] and SPH-based method[Petronetto et al. 2013] in a sphere diffusion example. We show the diffusion result at 0.5s in (a). In (b), the curves in light color denote the maximum absolute error, and the curves in dark color denote the L1-norm error.

11 DISCUSSION AND CONCLUSION

We propose a novel mesh-free simulation infrastructure to simulate codimensional surface tension phenomena. Our method models fluid features with volumes, thin sheets, filaments, and droplets, in different codimensions without requiring the tedious mesh evolution. At the heart of our approach is a set of codimensional MLS particles that can accommodate the numerical differentiation on the codimensional particle surfaces, thin threads, and the monolithic coupling across different codimensions. To the best of our knowledge, this numerical infrastructure is the first mesh-free Lagrangian solver that supports the codimensional PDE solvers and the non-manifold geometric transitions without maintaining explicit mesh connectivities.

From a Eulerian-Lagrangian perspective, our codimensional MLS method that couples particles and a background grid enables an accurate way to capture the intricate interfacial phenomena driven by strong surface tension by enhancing a standard PIC/FLIP solver with a monolithic solid-fluid coupling framework. From a purely Lagrangian point of view, our MLS particle approximation for thin sheets and filaments enables a connectivity-free approach to track the complex topological changes and differentiate physics on a non-manifold structure. The codimensional MLS particles, which can be regarded as a loosely defined Lagrangian mesh, provide nontrivial flexibility in handling topological and geometric evolutions, thanks to their particle nature. In this sense, our approach bridges the subareas of Lagrangian and Eulerian simulations by providing a unified approach that can start from either side and land on the other. Furthermore, the inherent connection of our codimensional particles with the point-based surface (in particular [Alexa et al. 2001]) in computer graphics bridges the communities of point cloud processing, rendering, and physics simulation, opening up new possibilities on the creation of novel dynamic simulation infrastructures by borrowing ideas from static geometry processing and data visualization.

One of the main limitations, which is also an interesting direction for our future work, is to reformulate the MLS operator to be symmetric. This reformulation can further lead to a positive-definite symmetric system for the monolithic coupling system to access the widely available high-performance multigrid solvers for large-scale simulations. The asymmetric nature of a particle MLS kernel has been a longstanding challenge in the point cloud processing community [Liang and Zhao 2013], with its codimensional variation to be even more challenging and promising. Second, our framework cannot simulate geometries exhibiting topological complexities, such as the Plateau borders of bubble clusters, due to the lack of effective representations of the codimensional topological connectivities on particles. This makes the work to develop point-based approaches that can capture more intricate codimensional transitions and explore the various fluid structural stabilization problems another interesting avenue for future work. Third, the meshless nature of our Lagrangian surface makes it challenging to distinguish interfaces that are close to each other, which could result in an incorrect codimensional transition or particle reseeding. Therefore, this approach is not suitable for modeling phenomena consisting of complex evolving thin sheets with frequent self collisions and merges. A more temporally consistent Lagrangian interface tracking scheme is needed to track multiple thin features that are interacting with each other. Last, our decoupled viscosity scheme simplifies the boundary conditions, which results in a Poisson-like linear system while introducing errors at the boundary. We plan to further investigate the fully coupled viscosity scheme that allows better modeling of viscosity with accurate boundary conditions.

For future work, we plan to explore the unified MLS scheme for particle fluids. We currently opt to use PIC/FLIP for codimension-0 simulation because we observed that particle-grid interpolation exhibits performance merits compared with particle least-squares fitting. Also, solving the projection step on a background grid allows the fewer number of degrees of freedom to describe the motion of a large bulk of liquids. On another front, unifying the projection step with an MLS expression is a valuable future direction to explore, following the existing literature (e.g., [Sin et al. 2009]) in this area. Specifically, we want to emphasize the possibility and the practicality to develop a particle-only variant by formulating the volumetric differential operators also using particle MLS in a Cartesian frame, as shown in Equation (2). Also, the meshless Lagrangian nature of our numerical infrastructure opens up possibilities in exploring structural design problems with thin sheets and filaments as the design constraints. We aim to develop this codimensional MLS particle tool to accommodate the challenging computational problems in a broad spectrum of areas in science and engineering to handle codimensional simulations and optimizations.

ACKNOWLEDGMENTS

We thank all the reviewers for their valuable comments. We want to thank valuable discussions with Dr. Shiying Xiong from Dartmouth College. This work was partially supported by the National Key Research and Development Program of China (2018YFB1004902) and the National Natural Science Foundation of China (61772329, 61373085). Bo Zhu acknowledges the funding support from Neukom Institute CompX Faculty Grant, Burke Research Initiation Award, Toyota TEMA North America Inc, ByteDance Inc, and NSF 1919647. We credit the Houdini Education licenses for the video generations.

REFERENCES

Bart Adams, Mark Pauly, Richard Keiser, and Leonidas J Guibas. 2007. Adaptively sampled particle fluids. In *ACM SIGGRAPH 2007 papers*. 48–es.

Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2013. Versatile surface tension and adhesion for SPH fluids. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 182.

Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T Silva. 2001. Point set surfaces. In *Proceedings of the Conference on Visualization'01*. IEEE Computer Society, 21–28.

Ryoichi Ando, Nils Thurey, and Reiji Tsuruno. 2012. Preserving fluid sheets with adaptively sampled anisotropic particles. *IEEE Transactions on Visualization and Computer Graphics* 18, 8 (2012), 1202–1214.

Stefan Auer, Colin B Macdonald, Marc Treib, Jens Schneider, and Rüdiger Westermann. 2012. Real-time fluid effects on surfaces using the closest point method. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 1909–1923.

Stefan Auer and Rüdiger Westermann. 2013. A Semi-Lagrangian Closest Point Method for Deforming Surfaces. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 207–214.

Stefan Band, Christoph Gissler, Andreas Peer, and Matthias Teschner. 2018. MLS pressure extrapolation for the boundary handling in divergence-free SPH. In *Proceedings of the 14th Workshop on Virtual Reality Interactions and Physical Simulations*. Eurographics Association, 55–63.

Stefan Band, Christoph Gissler, and Matthias Teschner. 2017. Moving least squares boundaries for SPH fluids. In *Proceedings of the 13th Workshop on Virtual Reality Interactions and Physical Simulations*. Eurographics Association, 21–28.

Christopher Batty and Robert Bridson. 2008. Accurate Viscous Free Surfaces for Buckling, Coiling, and Rotating Liquids.. In *Symposium on Computer Animation*. 219–228.

Christopher Batty, Andres Uribe, Basile Audoly, and Eitan Grinspun. 2012. Discrete viscous sheets. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 113.

Markus Becker and Matthias Teschner. 2007. Weakly compressible SPH for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 209–217.

Mikhail Belkin and Partha Niyogi. 2008. Towards a theoretical foundation for Laplacian-based manifold methods. *J. Comput. System Sci.* 74, 8 (2008), 1289–1308.

Mikhail Belkin, Jian Sun, and Yusu Wang. 2009. Constructing Laplace operator from point clouds in R d. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1031–1040.

Landon Boyd and Robert Bridson. 2012. MultiFLIP for energetic two-phase fluid simulation. *ACM Transactions on Graphics (TOG)* 31, 2 (2012), 16.

Jeremiah U Brackbill, Douglas B Kothe, and Hans M Ruppel. 1988. FLIP: a low-dissipation, particle-in-cell method for fluid flow. *Computer Physics Communications* 48, 1 (1988), 25–38.

Jeremiah U Brackbill and Hans M Ruppel. 1986. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational physics* 65, 2 (1986), 314–343.

Tyson Brochu and Robert Bridson. 2009. Robust topological operations for dynamic explicit surfaces. *SIAM Journal on Scientific Computing* 31, 4 (2009), 2472–2493.

Robbie Buckingham and John WM Bush. 2001. Fluid polygons. *Physics of Fluids* 13, 9 (2001), S10–S10.

John WM Bush and Alexander E Hasha. 2004. On the collision of laminar jets: fluid chains and fishbones. *Journal of fluid mechanics* 511 (2004), 285–310.

Jumyung Chang, Fang Da, Eitan Grinspun, and Christopher Batty. 2019. A Unified Simplicial Model for Mixed-Dimensional and Non-Manifold Deformable Elastic Objects. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2, 2 (2019), 1–18.

Ka Chun Cheung, Leevan Ling, and Steven J Ruuth. 2015. A localized meshless method for diffusion on folded surfaces. *J. Comput. Phys.* 297 (2015), 194–206.

Christophe Clanet. 2007. Waterbells and liquid sheets. *Annu. Rev. Fluid Mech.* 39 (2007), 469–496.

Georges-Henri Cottet, Petros D Koumoutsakos, Petros D Koumoutsakos, et al. 2000. *Vortex methods: theory and practice*. Cambridge university press.

Keenan Crane. 2018. Discrete differential geometry: An applied introduction. *Notices of the AMS, Communication* (2018), 1153–1159.

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 Transactions on Graphics (TOG)* 34, 4 (2015), 149.

Fang Da, David Hahn, Christopher Batty, Chris Wojtan, and Eitan Grinspun. 2016. Surface-only liquids. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 78.

Douglas Enright, Ronald Fedkiw, Joel Ferziger, and Ian Mitchell. 2002. A hybrid particle level set method for improved interface capturing. *Journal of Computational physics* 183, 1 (2002), 83–116.

Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. 2001. Visual simulation of smoke. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 15–22.

Florian Ferstl, Ryoichi Ando, Chris Wojtan, Rüdiger Westermann, and Nils Thuerey. 2016. Narrow band FLIP for liquid simulations. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 225–232.

Ming Gao, Andre Pradhana Tampubolon, Chenfanfu Jiang, and Eftychios Sifakis. 2017. An adaptive generalized interpolation material point method for simulating elastoplastic materials. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 223.

Robert A Gingold and Joseph J Monaghan. 1977. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly notices of the royal astronomical society* 181, 3 (1977), 375–389.

Qi Guo, Xuchen Han, Chuyuan Fu, Theodore Gast, Rasmus Tamstorf, and Joseph Teran. 2018. A material point method for thin shells with frictional contact. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–15.

Francis H Harlow. 1962. *The particle-in-cell method for numerical solution of problems in fluid dynamics*. Technical Report. Los Alamos Scientific Lab., N. Mex.

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.

Xiaowei He, Huamin Wang, Fengjun Zhang, Hongan Wang, Guoping Wang, and Kun Zhou. 2014. Robust simulation of sparsely sampled thin features in SPH-based free surface flows. *ACM Transactions on Graphics (TOG)* 34, 1 (2014), 7.

Jeong-Mo Hong, Ho-Young Lee, Jong-Chul Yoon, and Chang-Hun Kim. 2008a. Bubbles alive. *ACM Transactions on Graphics (TOG)* 27, 3 (2008), 48.

Jeong-Mo Hong, Jong-Chul Yoon, and Chang-Hun Kim. 2008b. Divergence-constrained moving least squares for fluid simulation. *Computer Animation and Virtual Worlds* 19, 3-4 (2008), 469–477.

Xiang Yu Hu and Nikolaus A Adams. 2006. A multi-phase SPH method for macroscopic and mesoscopic flows. *J. Comput. Phys.* 213, 2 (2006), 844–861.

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. 2018. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 150.

Markus Huber, Stefan Reinhardt, Daniel Weiskopf, and Bernhard Eberhardt. 2015. Evaluation of Surface Tension Models for SPH-Based Fluid Animations Using a Benchmark Test.. In *VRIPHYS*. 41–50.

Markus Ihmsen, Jens Cornelis, Barbara Solenthaler, Christopher Horvath, and Matthias Teschner. 2013. Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics* 20, 3 (2013), 426–435.

Sadashige Ishida, Masafumi Yamamoto, Ryoichi Ando, and Toshiya Hachisuka. 2017. A hyperbolic geometric flow for evolving films and foams. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 199.

Chenfanfu Jiang, Theodore Gast, and Joseph Teran. 2017. Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–14.

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015a. The affine particle-in-cell method. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 51.

Min Jiang, Yahan Zhou, Rui Wang, Richard Southern, and Jian Jun Zhang. 2015b. Blue noise sampling using an SPH-based method. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 211.

Ben Jones, Stephen Ward, Ashok Jallepalli, Joseph Perenia, and Adam W Bargteil. 2014. Deformation embedding for point-based elastoplastic simulation. *ACM Transactions on Graphics (TOG)* 33, 2 (2014), 21.

Jürgen Jost. 2008. *Riemannian geometry and geometric analysis*. Vol. 42005. Springer.

Theodore Kim, Jerry Tessendorf, and Nils Thuerey. 2013. Closest point turbulence for liquid surfaces. *ACM Transactions on Graphics (TOG)* 32, 2 (2013), 15.

Rongjie Lai, Jiang Liang, and Hongkai Zhao. 2013. A local mesh method for solving PDEs on point clouds. *Inverse Problems & Imaging* 7, 3 (2013).

Peter Lancaster and Kes Salkauskas. 1981. Surfaces generated by moving least squares methods. *Mathematics of computation* 37, 155 (1981), 141–158.

Egor Larionov, Christopher Batty, and Robert Bridson. 2017. Variational stokes: a unified pressure-viscosity solver for accurate viscous liquids. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–11.

Christian Ledergerber, Gaël Guennebaud, Miriah Meyer, Moritz Bächer, and Hanspeter Pfister. 2008. Volume MLS ray casting. *IEEE Transactions on Visualization and Computer Graphics* 14, 6 (2008), 1372–1379.

Shingyu Leung, John Lowengrub, and Hongkai Zhao. 2011. A grid based particle method for solving partial differential equations on evolving surfaces and modeling high order geometrical motion. *J. Comput. Phys.* 230, 7 (2011), 2540–2561.

Shingyu Leung and Hongkai Zhao. 2009a. A grid based particle method for evolution of open curves and surfaces. *J. Comput. Phys.* 228, 20 (2009), 7706–7728.

Shingyu Leung and Hongkai Zhao. 2009b. A grid based particle method for moving interface problems. *J. Comput. Phys.* 228, 8 (2009), 2993–3024.

Randall J LeVeque. 1996. High-resolution conservative algorithms for advection in incompressible flow. *SIAM J. Numer. Anal.* 33, 2 (1996), 627–665.

David Levin. 1998. The approximation power of moving least-squares. *Mathematics of computation* 67, 224 (1998), 1517–1531.

Jian Liang, Rongjie Lai, Tsz Wai Wong, and Hongkai Zhao. 2012. Geometric understanding of point clouds using Laplace-Beltrami operator. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 214–221.

Jian Liang and Hongkai Zhao. 2013. Solving partial differential equations on point clouds. *SIAM Journal on Scientific Computing* 35, 3 (2013), A1461–A1486.

Sebastian Martin, Peter Kaufmann, Mario Botsch, Eitan Grinspun, and Markus Gross. 2010. Unified simulation of elastic rods, shells, and solids. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 1–10.

Niloy J Mitra and An Nguyen. 2003. Estimating surface normals in noisy point cloud data. In *Proceedings of the nineteenth annual symposium on Computational geometry*. ACM, 322–328.

Joe J Monaghan. 1992. Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics* 30, 1 (1992), 543–574.

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*. Eurographics Association, 154–159.

Matthias Müller, Richard Keiser, Andrew Nealen, Mark Pauly, Markus Gross, and Marc Alexa. 2004. Point based animation of elastic, plastic and melting objects. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 141–151.

Andrew Nealen. 2004. An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered data approximation and interpolation. *URL: http://www. nealen. com/projects* 130, 150 (2004), 25.

Mark Pauly, Richard Keiser, Bart Adams, Philip Dutré, Markus Gross, and Leonidas J Guibas. 2005. Meshless animation of fracturing solids. In *ACM Transactions on Graphics (TOG)*, Vol. 24. ACM, 957–964.

Andreas Peer and Matthias Teschner. 2016. Prescribed velocity gradients for highly viscous SPH fluids with vorticity diffusion. *IEEE transactions on visualization and computer graphics* 23, 12 (2016), 2656–2662.

Fabiano Petronetto, Afonso Paiva, Elias S Helou, DE Stewart, and Luis Gustavo Nonato. 2013. Mesh-Free Discrete Laplace–Beltrami Operator. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 214–226.

Nick Rasmussen, Douglas Enright, Duc Nguyen, Sebastian Marino, Nigel Sumner, Willi Geiger, Samir Hoon, and Ronald Fedkiw. 2004. Directable photorealistic liquids. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*. 193–202.

Avi Robinson-Mosher, Craig Schroeder, and Ronald Fedkiw. 2011. A symmetric positive definite formulation for monolithic fluid structure interaction. *J. Comput. Phys.* 230, 4 (2011), 1547–1566.

Steven J Ruuth and Barry Merriman. 2008. A simple embedding method for solving partial differential equations on surfaces. *J. Comput. Phys.* 227, 3 (2008), 1943–1961.

Robert Saye. 2014. High-order methods for computing distances to implicitly defined surfaces. *Communications in Applied Mathematics and Computational Science* 9, 1 (2014), 107–141.

Frank Schmidt. 2014. The laplace-beltrami-operator on riemannian manifolds. In *Seminar Shape Analysis*.

Craig Schroeder, Wen Zheng, and Ronald Fedkiw. 2012. Semi-implicit surface tension formulation with a Lagrangian surface mesh on an Eulerian simulation grid. *J. Comput. Phys.* 231, 4 (2012), 2092–2115.

G Shobeyri and MH Afshar. 2010. Simulating free surface problems using discrete least squares meshless method. *Computers & Fluids* 39, 3 (2010), 461–470.

Funshing Sin, Adam W Bargteil, and Jessica K Hodgins. 2009. A point-based method for animating incompressible flow. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 247–255.

Tetsuya Takahashi, Yoshinori Dobashi, Issei Fujishiro, Tomoyuki Nishita, and Ming C Lin. 2015. Implicit formulation for SPH-based viscous fluids. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 493–502.

Alexandre Tartakovsky and Paul Meakin. 2005. Modeling of surface tension and contact angles with smoothed particle hydrodynamics. *Physical Review E* 72, 2 (2005), 026301.

Nils Thürey, Chris Wojtan, Markus Gross, and Greg Turk. 2010. A multiscale approach to mesh-based surface tension flows. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 48.

Henk A Van der Vorst. 1992. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on scientific and Statistical Computing* 13, 2 (1992), 631–644.

Xiao-Kun Wang, Xiao-Juan Ban, Ya-Lan Zhang, Si-Nuo Liu, and Peng-Fei Ye. 2017. Surface tension model based on implicit incompressible smoothed particle hydrodynamics for fluid simulation. *Journal of Computer Science and Technology* 32, 6 (2017), 1186–1197.

Nicholas J Weidner, Kyle Piddington, David IW Levin, and Shinjiro Sueda. 2018. Eulerian-on-lagrangian cloth simulation. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–11.

Marcel Weiler, Dan Koschier, Magnus Brand, and Jan Bender. 2018. A physically consistent implicit viscosity solver for SPH fluids. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 145–155.

Rene Winchenbach, Hendrik Hochstetter, and Andreas Kolb. 2017. Infinite continuous adaptivity for incompressible SPH. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–10.

Jihun Yu and Greg Turk. 2010. Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 217–225.

Yonghao Yue, Breannan Smith, Christopher Batty, Changxi Zheng, and Eitan Grinspun. 2015. Continuum foam: A material point method for shear-dependent flows. *ACM Transactions on Graphics (TOG)* 34, 5 (2015), 160.

Mingyu Zhang. 2010. Simulation of surface tension in 2D and 3D with smoothed particle hydrodynamics method. *J. Comput. Phys.* 229, 19 (2010), 7238–7259.

Yizhong Zhang, Huamin Wang, Shuai Wang, Yiying Tong, and Kun Zhou. 2011. A deformable surface model for real-time water drop animation. *IEEE Transactions on Visualization and Computer Graphics* 18, 8 (2011), 1281–1289.

Wen Zheng, Jun-Hai Yong, and Jean-Claude Paul. 2006. Simulation of bubbles. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 325–333.

Wen Zheng, Bo Zhu, Byungmoon Kim, and Ronald Fedkiw. 2015. A new incompressibility discretization for a hybrid particle MAC grid representation with surface tension. *J. Comput. Phys.* 280 (2015), 96–142.

Bo Zhu, Minjae Lee, Ed Quigley, and Ronald Fedkiw. 2015. Codimensional non-Newtonian fluids. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 115.

Bo Zhu, Ed Quigley, Matthew Cong, Justin Solomon, and Ronald Fedkiw. 2014. Codimensional surface tension flow on simplicial complexes. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 111.

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 965–972.

A WEIGHTED PCA

To obtain the local frame $\mathbf{F}_p$ on a particle, we employ the weighted version of principal component analysis (WPCA) [Yu and Turk 2010] to robustly generate the three frame axes from a discrete point cloud. For a particle $\vec{x}_p$ surrounded by a set of neighboring particles $\mathcal{N}_p = \{q \mid ||\vec{x}_p - \vec{x}_q|| < h\}$, with h as the local supporting radius, the covariance matrix $\mathbf{C}_p$ is computed as:

$$\mathbf{C}_p = \frac{\sum_q \Phi_{pq}(\vec{x}_q - \vec{x}_p^w)(\vec{x}_q - \vec{x}_p^w)^T}{\sum_q \Phi_{pq}}, \tag{25}$$

$$\vec{x}_p^w = \frac{\sum_q \Phi_{pq}\vec{x}_q}{\sum_q \Phi_{pq}} \tag{26}$$

with the weighting function Φ defined as

$$\Phi_{pq} = \begin{cases} 1 - (\frac{||\vec{x}_p - \vec{x}_q||}{h})^3, if \ ||\vec{x}_p - \vec{x}_q|| < h \\ 0, otherwise \end{cases} \tag{27}$$

We obtain the eigenvectors and eigenvalues from the covariance matrix by performing singular value decomposition (SVD):

$$\mathbf{C}_p = \mathbf{R}\Sigma\mathbf{R}^T, \tag{28}$$

$$\Sigma = \begin{bmatrix} b^1 & & \\ & b^2 & \\ & & b^3 \end{bmatrix} \tag{29}$$

where $\mathbf{R}$ is a rotation matrix with principal axes as column vectors. The resulting eigenvectors $\vec{b}^1$,$\vec{b}^2$,$\vec{b}^3$ of the covariance matrix $\mathbf{C}$ give the three orthonormal basis vectors of the local frame $\mathbf{F}_p$, corresponding to the eigenvalues b^1, b^2, b^3 (sorted by descending order).

B MLS

We construct the codimensional differential operators using MLS following the work of [Liang and Zhao 2013]. The procedure consists of 3 steps.

Step I: Local frame. The local frame $\mathbf{F}_p$ is built on each particle following Appendix A. Specifically, in codimension-1, $\vec{b}^3$ describes the surface normal and $\vec{b}^1$, $\vec{b}^2$ are the tangent vectors in the local plane, while in codimension-2, $\vec{b}^1$, $\vec{b}^2$, $\vec{b}^3$ are the tangent, normal, and binormal vector of the filament respectively.

Step II: Local function approximation. Consider a codimension-1 particle p, the local position of a neighboring particle q in its local frame is denoted as $(\xi_q^1, \xi_q^2, \xi_q^3)$. With the MLS method, the underlying surface (i.e., a continuous approximation of ξ^3) can be approximated with a bivariate quadratic function

$$\hat{\xi}^3(\boldsymbol{\beta}, \xi^1, \xi^2) = \beta_1 + \beta_2\xi^1 + \beta_3\xi^2 + \beta_4(\xi^1)^2 + \beta_5(\xi^1\xi^2) + \beta_6(\xi^2)^2 \tag{30}$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, ..., \beta_6]^T$ is the coefficient vector.

The approximation is obtained by minimizing the weighted sum of the squared errors on each particle q as

$$\min_{\boldsymbol{\beta}} \sum_q \Psi(d_q)(\hat{\xi}^3(\boldsymbol{\beta}, \xi_q^1, \xi_q^2) - \xi_q^3)^2. \tag{31}$$

We adopt the weighting function introduced in [Liang et al. 2012]

$$\Psi(d) = \begin{cases} 1, & d = 0 \\ 1/n, & d > 0 \text{ and } d \le h \\ 0, & d > h \end{cases}. \tag{32}$$

By further defining $\boldsymbol{\xi}^3 = [\xi_1^3, \xi_2^3, ..., \xi_n^3]^T$, the vectorized form of Equation (31) becomes

$$\min_{\boldsymbol{\beta}}(\mathbf{B}\boldsymbol{\beta} - \boldsymbol{\xi}^3)^T\mathbf{W}(\mathbf{B}\boldsymbol{\beta} - \boldsymbol{\xi}^3). \tag{33}$$

Here $\mathbf{B}$ is an $n \times 6$ matrix with each row as the polynomial basis vector of a neighboring particle q

$$\mathbf{B} = \begin{bmatrix} 1 & \xi_0^1 & \xi_0^2 & (\xi_0^1)^2 & (\xi_0^1\xi_0^2) & (\xi_0^2)^2 \\ 1 & \xi_1^1 & \xi_1^2 & (\xi_1^1)^2 & (\xi_1^1\xi_1^2) & (\xi_1^2)^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \xi_n^1 & \xi_n^2 & (\xi_n^1)^2 & (\xi_n^1\xi_n^2) & (\xi_n^2)^2 \end{bmatrix}.$$

The matrix $\mathbf{W}$ is an $n \times n$ diagonal with q-th diagonal element as the weight $\Psi(r_q)$. The solution of (33) is given as $\boldsymbol{\beta} = (\mathbf{B}^T\mathbf{W}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{W}\boldsymbol{\xi}^3$. By letting $\mathbf{H} = (\mathbf{B}^T\mathbf{W}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{W}$, the solution can be simplified as

$$\boldsymbol{\beta} = \mathbf{H}\boldsymbol{\xi}^3 \tag{34}$$

Then, two tangent basis vectors are given by $\vec{t}^1 = (1, \frac{\partial\xi^3}{\partial\xi^1}, 0) = (1, \beta_2, 0)$ and $\vec{t}^2 = (1, 0, \frac{\partial\xi^3}{\partial\xi^2}) = (1, 0, \beta_3)$, and the metric tensor is

$$[g_{kl}] = \begin{bmatrix} \beta_2^2 + 1 & \beta_2\beta_3 \\ \beta_2\beta_3 & \beta_3^2 + 1 \end{bmatrix}.$$

To approximate a scalar field s, the same procedure is performed with the polynomial $\hat{s}(\boldsymbol{\gamma}, \xi^1, \xi^2)$, which yields $\boldsymbol{\gamma} = \mathbf{H}\mathbf{s}$. For a vector field $\vec{v}$, two scalar components of it are approximated separately. Note that $\mathbf{H}$ is reusable to fit any field on the same particle distribution.

Step III: Local Differential Operator. In the last step, we define the local differential operator on the codimension-1 surface for a scalar field s or a vector field $\vec{v}$. After the MLS fitting in Step II, the first and second order derivatives of s can be deduced easily, e.g., $\frac{\partial s}{\partial\xi^1} = \gamma_2$ and $\frac{\partial^2 s}{(\partial\xi^2)^2} = 2\gamma_6$. With the metric tensor and partial derivatives, we obtain the codimension-1 operators as Equation (7). To further construct the Laplacian matrix $\mathbf{L}_{(1)}$ used in Equation (13)(16), we write Equation (7c) as

$$\nabla_{(1)}^2 s_p = \mathbf{g}^T\mathbf{H}\mathbf{s}, \tag{35}$$

where $\mathbf{g} = [0, 0, 0, 2g^{11}, g^{12} + g^{21}, 2g^{22}]^T$. $\mathbf{g}^T\mathbf{H}$ is a $1 \times n$ row vector describing the contribution from neighboring particles. We extend $\mathbf{g}^T\mathbf{H}$ into a $1 \times N_{(1)}$ row vector by filling respective entries of non-neighboring particles with 0 and stack the row vectors of all codimension-1 particles to form the Laplacian matrix $\mathbf{L}_{(1)}$.

As for the codimension-2 filament parameterized with one parameter ξ^1, we can approximate its underlying manifold using two quardratic functions $\hat{\xi}^2(\boldsymbol{\gamma}, \xi^1)$, $\hat{\xi}^3(\boldsymbol{\delta}, \xi^1)$. The tangent basis vector is $\vec{t}^1 = (1, \partial\xi^2/\partial\xi^1, \partial\xi^3/\partial\xi^1) = (1, \gamma_1, \delta_1)$ and the 1×1 metric tensor is $[g] = [1 + \gamma_2^2 + \delta_2^2]$. The differential operators are then built in the same way.

C BOUNDARY CONDITION

The free surface boundary is treated by different codimensional groups implicitly. The solid boundary is represented using the standard cut-cell method in the projection step. Besides, in the blowing bubble and catenoid examples, we sampled the solid rings with static boundary particles, which are incorporated in the codimension-1 MLS solver.

D SURFACE AREA APPROXIMATION

As shown in Equation (17), the surface areas on codimension-1 and codimension-2 particles are necessary for calculating the surface tension force for each particle. For a codimension-1 particle, we estimate its surface area using a simplified version of the local Voronoi method (e.g., see [Lai et al. 2013]). For a thin film that is not the surface of a fluid volume, we double the surface area to consider the surface tension existing on both sides. For a codimension-2 particle, we treat its local geometry as a cylindrical tube with its radius as the particle radius r_p and the length l_p as the distance between the two closest particles and estimate its surface area as the lateral area of the tube. For a rim particle, we follow [Bush and Hasha 2004] to divide its surface tension into two components: the rim surface tension component with the surface area $S_{r,p} = 2r_p l_p$ equal to the sectional area of the tube, and the codimension-2 surface tension component where the surface area $S_{f,p} = 2(\pi - 1)r_p l_p$ is the lateral area of the tube subtracting the sectional area. And for a bud particle, the surface area used in the bud surface tension is $S_{b,p} = \pi r_p^2$.

E PHYSICAL PARAMETER

The density is 1000 kg/m^3. The resolutions and other parameters are listed in Table 1 and Table 2. In the blowing bubble example, the radius of the circle membrane is 0.0192m. In the bursting bubble example, the bubble radius is 0.0576m. In the catenoid example, the cylinder membrane is initialized with the length 0.0384m and the radius 0.0512m. We initialized the droplet with the radius 0.0128m in the droplet on the plane example. The cubic droplets with the length 0.024m are constructed in the droplet collision example. In the fluid chain and dripping water examples, we emit the jets with the radii 0.008m. In the waterbell example and fluid polygon example, the radii are 0.007m.

Scene	#grid res	#active cells	$\#C0$	$\#C1$	$\#C2$
Blowing bubble	-	-	0	239k	0
Film catenoid	-	-	0	63k	0
Bursting bubble	-	-	0	70k	2k
Droplet on the plane	64×64×64	1k	52k	9.7k	0
Droplet collision	64×64×64	4k	191k	20k	0
Dripping water	64×200×64	3k	63k	32k	0
Fluidchain	96×256×96	13k	222k	119k	0
Waterbell	192×192×192	14k	273k	163k	0
Polygon-6	192×72×192	2k	39k	157k	5k
Polygon-8	192×72×192	2k	37k	148k	4k

Table 1. Simulation resolutions in scenarios

Scene	$\Delta t(s)$	σ(N/m)	μ(mPa· s)	v_0(m/s)
Blowing bubble	0.005	0.01	0	0
Film catenoid	0.005	0.015	0	0
Bursting bubble	0.002	5×10^{-4}	0	0
Droplet on the plane	0.003	0.03	0.01-0.25	0
Droplet collision	0.005	0.03	0	±0.02
Dripping water	0.005	0.015-0.06	0	0.07
Fluidchain	0.005	0.045	0	0.2
Waterbell	0.002	0.0075	0.1	±0.12
Fluid Polygon	0.005	0.05	0	±0.15

Table 2. Physical parameters in scenarios

F PARTICLE EMISSION AND DISTRIBUTION

During the initialization step, the cubic droplets in the droplet collision example (see Figure 8) are sampled on a regular grid with the outermost layer composed of codimension-1 particles. The spherical droplet is constructed using equidistributed concentric sphere surfaces of equidistant radii with the initial particle spacing as the intervals. The equidistributed sphere is sampled by placing particles evenly on circles of latitude. The intervals between circles and the intervals between particles in longitude are the same as the initial particle spacing. We emit cylindrical jets using equidistributed circle patterns where the outline of the circle is sampled by codimension-1 particles in dripping water, fluid chain, waterbell, and fluid polygon examples. Other shapes, such as the circle membrane in the bubble examples and the cylinder membrane in the catenoid example, are sampled in a similar way. The particle properties, e.g., particle mass, are initialized uniformly during the emission step. A particle's radius is initialized according to its initial mass at the beginning and evolved accordingly afterward. The cell size for our simulation is $\Delta x = h$. The initial particle spacing is $\Delta x/4$ for the droplet collision and the droplet on the plane examples, and $\Delta x/3$ for other examples. We observed that a higher distribution density avoids missing the constraint faces where codimensional coupling happens and the interior cells where large deformation happens.