Tube Maps: Fast SPH Boundary Handling with Tubular Coordinates

DARIA NOGINA, Columbia University, USA
SILVIA SELLÁN, Columbia University, USA

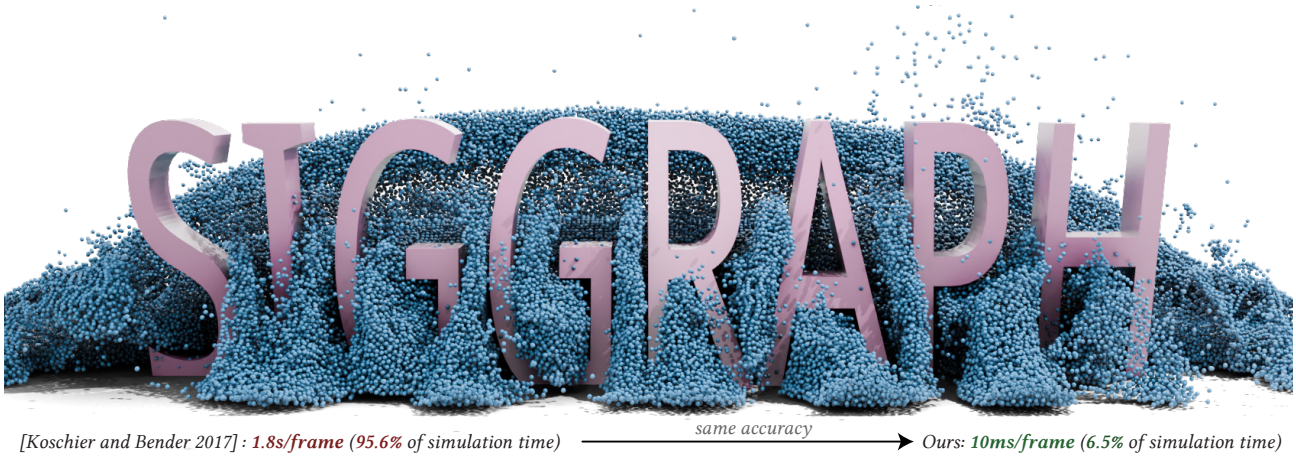


Fig. 1. In an SPH fluid simulation with 400,000 particles and several deforming solids, resolving fluid-solid interactions can become a computational bottleneck. We propose a drop-in replacement for existing implicit boundary handling methods that reduces their runtime by several orders of magnitude.

Smoothed Particle Hydrodynamics (SPH) simulations rely on accurately and efficiently modeling fluid-solid interactions. However, particle-based coupling strategies introduce non-deterministic discretization errors, and implicit methods achieve high accuracy at the cost of expensive numerical integration. We introduce Tube Maps, a drop-in replacement for SPH boundary density computation that achieves accuracy comparable to implicit methods while dramatically reducing their computational cost. Our key observation is that the boundary density integral is fully determined by the local surface geometry near a fluid particle's closest point. By expressing this geometry in tubular coordinates, we reduce the original three-dimensional integral to a one-dimensional closed-form expression that can be evaluated in constant time. We thus eliminate numerical quadrature and reduce boundary handling costs by one to three orders of magnitude, enabling fast and accurate SPH simulations with time-varying curved solids.

CCS Concepts: • **Computing methodologies** → **Physical simulation**; • **Theory of computation** → **Computational geometry**.

Additional Key Words and Phrases: Fluid Animation, Computational Geometry

ACM Reference Format:
Daria Nogina and Silvia Sellán. 2026. Tube Maps: Fast SPH Boundary Handling with Tubular Coordinates. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '26), July 19–23, 2026, Los Angeles, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3799902.3811118

Authors' Contact Information: Daria Nogina, Columbia University, New York, USA, nogina.daria.ml@gmail.com; Silvia Sellán, Columbia University, New York, USA, silviasellan@cs.columbia.edu.

SIGGRAPH Conference Papers '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2554-8/26/07
https://doi.org/10.1145/3799902.3811118

1 Introduction

Smoothed Particle Hydrodynamics (SPH) is a popular Lagrangian fluid simulation framework thanks to its flexibility, efficiency and visual realism. In applications from computer graphics to engineering, a fundamental part of any SPH simulation is the modeling of interactions between fluids and solids (e.g., characters, containers or mechanical devices). A particularly critical aspect of this solid *boundary handling* is the computation of the contribution of each solid to the density field at each fluid particle.

Unfortunately, existing methods provide a sharp tradeoff between simulation speed and accuracy. By far the most common tactic, popularized by Akinci et al. [2012], is to discretize the solid's surface using *boundary particles*, which contribute to the simulation in a manner analogous to their fluid counterparts but whose position is rigidly prescribed. As shown by Koschier and Bender [2017], this introduces non-deterministic discretization error which can only be alleviated via costly increases to sampling density.

To remedy this, Koschier and Bender [2017] and Bender et al. [2019] proposed *implicit* boundary handling methods. In these, the solid's contribution to the fluid's density is computed through a

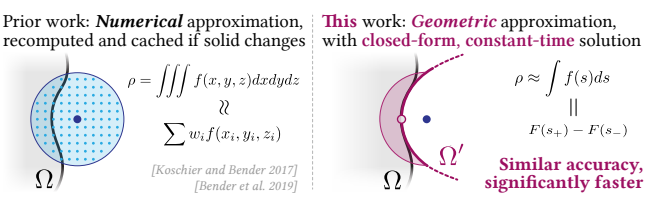


Fig. 2. Our algorithm's key insight is substituting the numerically approximated integral of prior work for a geometric approximation with a closed-form solution, massively improving runtime.

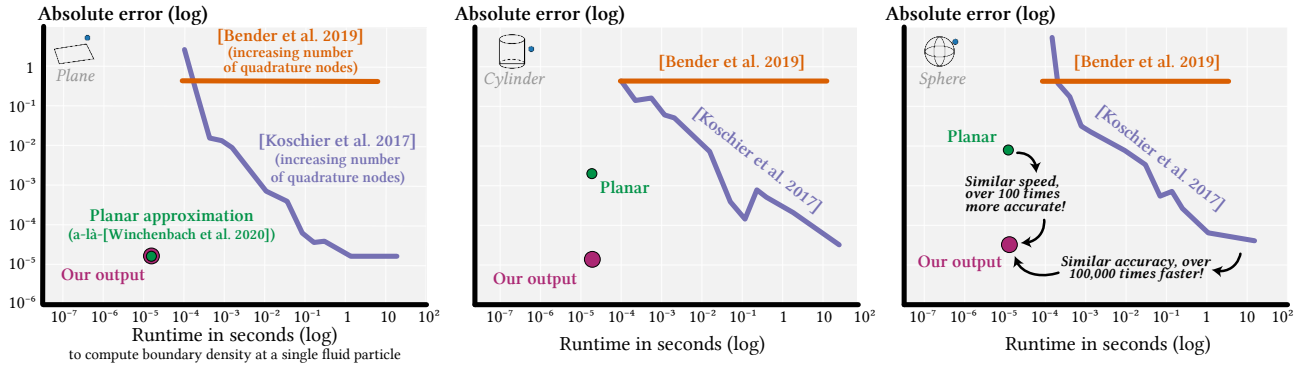


Fig. 3. Prior work on estimating boundary contributions to a fluid particle's density introduces a tradeoff between runtime and accuracy: planar approximations yield fast-yet-inaccurate results for curved geometry (green, center and right), while numerical methods can achieve high accuracy only by relying on an impossibly large number of quadrature evaluations (purple and orange lines). Our new implicit boundary handling strategy matches the accuracy of expensive numerical methods without their computational cost.

three-dimensional integral that relies on a signed distance representation of the object, greatly improving accuracy. This integral can be estimated analytically if the shape is planar or close to planar [Winchenbach et al. 2020]; however, numerically computing it for curved shapes (e.g., using quadrature) is computationally costly. In cases in which the solid shapes are changing over time (e.g., due to elasticity, prescribed motion or geometric optimization), this integral must be recomputed in each step of the simulation, which can quickly become a computational bottleneck (see Figure 1).

We introduce *Tube Maps*, a drop-in replacement for existing SPH boundary density computation that provides an accuracy similar to prior implicit coupling methods while reducing their runtime by several orders of magnitude (see Figure 1). Our key insight is that by proposing to approximate the integral numerically, previous works are agnostic to the form of the specific integrand. In particular, they do not take advantage of the fact that the integrand is fully determined by a shape's local geometry around a fluid particle's closest point (see Figure 2, left). Instead of this numerical approximation, we propose a surface-based *geometric* approximation that considers the surface's representation in tubular coordinates and converts the 3D integral into a one-dimensional one with a closed-form analytical solution that can be computed in constant time from the fluid particle's SDF and the surface's curvature (see Figure 2, right).

By substituting the hundreds of thousands of quadrature node evaluations of prior work with a single closed-form formula, our algorithm reduces the computational cost of SPH boundary handling by factors of 10 to 1000. Our contribution is most notable in scenes in which the solid's shape is complex and changes over time. In these simulations, the density integral cannot be pre-computed and must instead be approximated and cached with high grid resolution in every simulation step. For medium-scale scenes with hundreds of thousands of particles and deforming solids, our algorithm reduces the total simulation runtime by several orders of magnitude (see Figure 1). Unlike prior methods, which approximate the surface with planar patches to sacrifice fidelity for speed, our geometric approximation not only matches but often outperforms the accuracy of costly numerical quadrature (see Figure 3).

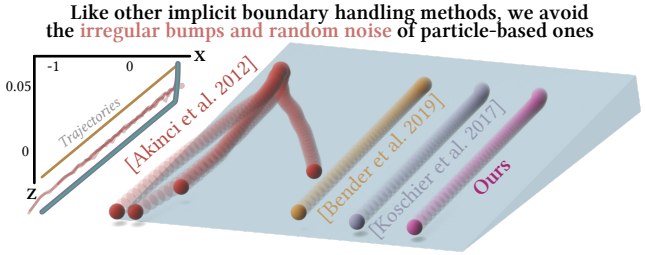


Fig. 4. A single particle rolls down an inclined plane in this example inspired by Koschier and Bender [2017]. Particle-based methods create irregular behaviour that is avoided by implicit boundary handling.

Through extensive quantitative experimentation on simple analytical geometries, we demonstrate our algorithm's superior accuracy and efficiency. We implement our algorithm in a fork of a popular open-source SPH simulation library [Bender 2025], and highlight its nature as a plug-in replacement for prior boundary handling strategies through a diverse set of prototypical applications with complex, curved geometries.

2 Related Work

Smoothed Particle Hydrodynamics (SPH) is an increasingly popular fluid simulation framework. Due to its meshless nature and relatively simple implementation, it has been applied in a wide array of scientific fields: from astrophysics [Gingold and Monaghan 1977] and aerospace [Siemann and Groenenboom 2014] and automotive engineering [Cleary et al. 2006; Fleissner et al. 2010] to marine industrial design [Le Touzé et al. 2010; Marrone et al. 2011] and geophysics [Gutfraind and Savage 1998].

In the computer graphics research community, SPH has grown in popularity thanks to its ability to produce visually plausible simulations of fluids in complex scenes. Over the past two decades, a significant amount of work has been dedicated to improving the applicability and robustness of SPH in graphics: by simulating turbulent flows [Launder and Spalding 1972], viscous [Weiler et al. 2018] and multi-phase fluids [Pozorski and Olejnik 2024], as well as

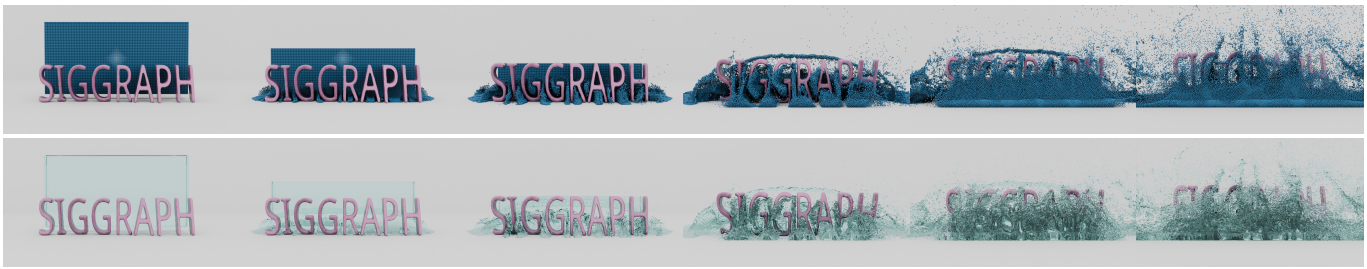


Fig. 5. A set of smooth, curved letters interact with a violent dam break (an invisible wall stands between them and the camera). See Supplemental Video.

through novel numerical techniques to enforce incompressibility [Gissler et al. 2019; Skillen et al. 2013]. In recent years, a growing body of work has considered strategies to augment SPH simulation using data-driven priors [Dou et al. 2025], and even designed end-to-end differentiable SPH simulation pipelines [Li et al. 2023; Winchenbach and Thuerey 2025].

In this paper, we concern ourselves with one specific step in most SPH pipelines: namely, the computation of the fluid's density in regions neighboring solid boundaries or *boundary handling*. Our work is thus orthogonal and complementary to the vast existing literature dedicated to expanding or improving other aspects of SPH simulation. Because of this, we will focus this section only on the discussion of prior SPH boundary handling strategies, and refer the reader to existing surveys [Ihmsen et al. 2014; Koschier et al. 2022] and courses [Koschier et al. 2020] for a comprehensive review of SPH in computer graphics.

2.1 SPH Boundary Handling

A critical element in SPH fluid–solid coupling is the estimation of field variables (e.g., density) at fluid particles near the solid-fluid interface. Mathematically, this amounts to computing the integral of a function over a complex volumetric domain: the intersection of the solid with the SPH kernel's support domain centered at the fluid particle. For this task, different strategies have been proposed: for our purposes, it is useful to characterize them *geometrically* based on how this complex volumetric domain is discretized.

Point-based or particle-based methods. The most common SPH boundary handling strategy consists of densely sampling the solid boundary with its own boundary particles, which are equipped with estimated pseudovolumes and contribute to the fluid's density in a way analogous to fluid particles [Akinci et al. 2012]. If the solid boundary is changing or deforming, the particles must be resampled to avoid leakage [Akinci et al. 2013].

Point-based methods are fast and flexible; however, they introduce artificial irregularities on the solid's boundary that can cause non-deterministic artifacts and friction (see Figure 4). If no additional considerations, such as those in [Band et al. 2018], are made, this drawback can only be alleviated by increasing the density of the sampling, leading to a significant computational cost even for the simplest boundary geometries.

When the boundary surface is flat, or its curvature is low enough, *planar methods* estimate its contribution to the fluid density can be estimated analytically. For example, Winchenbach et al. [2020]

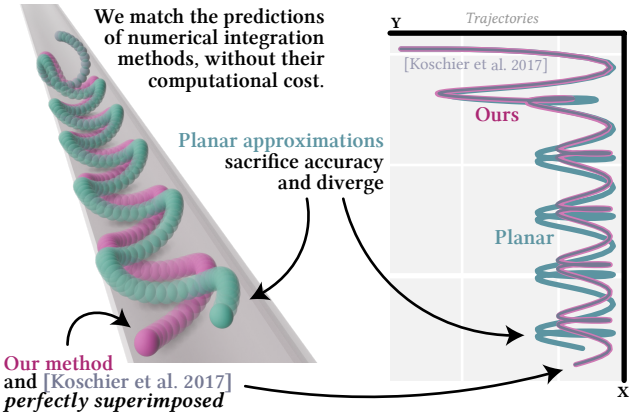


Fig. 6. A particle rolls down a cylinder using our method, the density maps proposed by Koschier and Bender [2017], and a planar approximation in the style of Winchenbach et al. [2020]. The latter accumulates error and diverges, while ours matches numerical integration without its cost.

approximate the boundary by a plane defined by the surface normal at the closest point to the particle. Planar approximation allows for a sufficient simplification of the domain to derive a fully analytical solution for a 2D case [Winchenbach and Kolb 2025]. On the other hand, *multi-planar methods* also rely on flat primitives, but in a different sense: they do not replace the local boundary by one tangent plane but integrate over a piecewise-flat surface description. Fujisawa and Miura [2015] refine the estimate by progressively clipping the domain with additional planes at the cost of increased runtime.

The main drawback of planar and multiplanar methods is fundamental: by approximating the surface with flat elements, their accuracy depend on the relation between the surface's curvature and the SPH kernel's support radius. When the solid is flat (see Figure 3, left) or the particle radius is very small, they provide a fast and accurate estimation. However, for moderately curved solids or coarse particles, they quickly accumulate error and visibly deviate from analytical solutions (see Figure 6). Inspired by these, our work introduces a *second-order* geometric approximation of the solid's boundary, which not only matches the accuracy of these methods when the surfaces are flat but is also orders of magnitude more accurate for arbitrarily curved boundaries (see Figure 3, right).

Numerical methods. Rather than relying on elaborate geometric representations, approaches like *Density Maps* proposed by Koschier

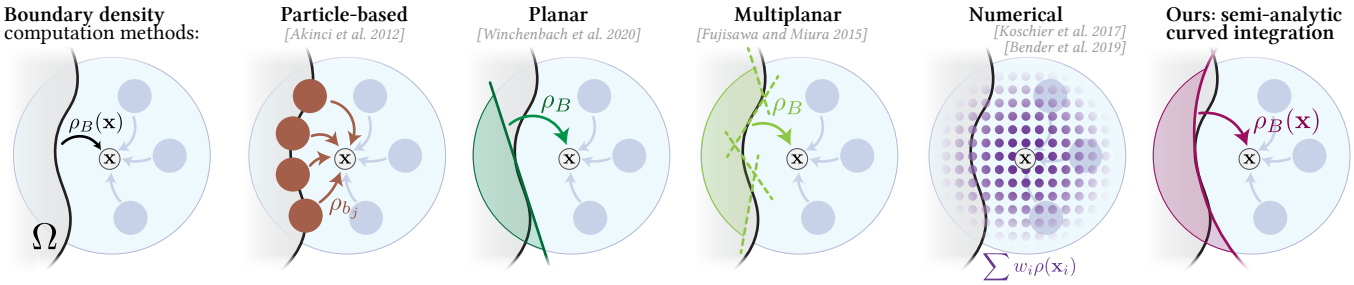


Fig. 7. A review of strategies to correct boundary density underestimation in SPH: from particle-based methods to planar and multi-planar estimations, numerical methods and our novel curved geometric approximation.

and Bender [2017] estimate boundary density via numerical integration schemes such as Gauss–Legendre quadrature. Later, Bender et al. [2019] proposed *Volume Maps*, which separate the boundary density integral into two terms that are estimated separately, improving numerical stability at the cost of accuracy.

Numerical methods avoid the irregularities of particle-based approaches and significantly outperform planar approximations for arbitrarily curved boundaries. The volumetric nature of their integration allows them to recover missing kernel support from information in the solid volume. However, they do so at a significant computational cost, requiring hundreds of thousands of quadrature node evaluations for a single spatial boundary density evaluation. These values may be pre-computed and cached at the beginning of a simulation if the solid is rigid [Koschier et al. [n. d.]; Koschier and Bender 2017], but must be re-computed fully or partially [Ma et al. 2025] at every simulation frame if the solid is deforming or animated. Our *Tube Maps* approach builds directly on these methods; critically, we provide an elaborate mathematical derivation that allows us to approximate the same smoothed integral proposed by Koschier and Bender [2017] in a surface-based manner - *geometrically* instead of numerically, leading to a constant-time drop-in replacement that improves their runtime and accuracy by orders of magnitude in the presence of curved solids (see Figure 3). In section 3, we review these strategies in more detail and set the stage for our main contribution.

3 Background: SPH and SPH Boundary Handling

The state of an incompressible fluid is usually expressed in terms of its velocity $\mathbf{v}(\mathbf{x})$, density $\rho(\mathbf{x})$ and pressure $p(\mathbf{x})$ functions. These are governed by the Eulerian *Navier-Stokes equations*

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\mathbf{v} = -\frac{1}{\rho}\nabla p + \mu\nabla^2\mathbf{v} + g\,, \qquad \nabla\cdot\mathbf{v} = 0\,,$$

where g represents external forces (e.g., gravity) and μ is the fluid's viscosity coefficient. Intuitively, its first equality says that the fluid's velocity changes due to advection, pressure, viscosity and external forces, and the second ensures incompressibility.

A common reformulation of these equations is to imagine $\mathbf{x}$ not as a fixed position in space but as the trajectory $\mathbf{x}(t)$ of a fluid particle moving with the fluid flow, i.e.,

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}). \tag{1}$$

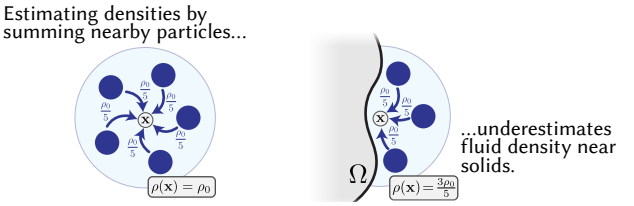


Fig. 8. Correct fluid density estimation in SPH requires boundary handling.

In this case, the Navier-Stokes equations can be rewritten as

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho}\nabla p + \mu\nabla^2\mathbf{v} + g\,, \quad \nabla\cdot\mathbf{v} = 0\,, \tag{2}$$

known as the *Lagrangian* Navier-Stokes equations.

3.1 Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) is a framework encompassing a broad set of methods for solving the Lagrangian Navier-Stokes equations. They start by considering N particle trajectories $\mathbf{x}_i(t)$ with initial positions $\mathbf{x}_i(0)$, each with its own mass m_i and state variables $\mathbf{v}_i(t)$, $\rho_i(t)$ and $p_i(t)$ governed by Equations 1 and 2.

Then, for each particle i surrounded by other fluid particles, SPH discretizes some or all of these state variables (collectively denoted a_i) as a weighted average of the state at its neighbors. Concretely, using a compactly supported kernel W_h with support radius h, and letting $\mathcal{N}(i)$ be the set of fluid particles in a neighborhood of radius h around $\mathbf{x}_i$, they set

$$a_i = \sum_{j\in\mathcal{N}(i)} \frac{m_j}{\rho_j} a_j W_h(\mathbf{x}_i - \mathbf{x}_j)\,. \tag{3}$$

Conveniently, this discretization allows one to write the differential quantities of a_i (e.g., its gradient, ∇a_i, divergence $\nabla\cdot a_i$, and Laplacian $\nabla^2 a_i$) in terms of the spatial derivatives of W_h and the state of the neighboring particles a_j.

Different ways of exploiting this fact lead to different strategies for solving Equation 2. Most employ operator splitting: at each simulation step, the viscosity and external forces are first used to obtain a preliminary velocity. Then, the fluid density ρ_i is calculated at each particle, which is used to estimate its pressure p_i (be it via an equation of state [Becker and Teschner 2007], iteratively [Bender and Koschier 2015; Solenthaler and Pajarola 2009] or with a PDE

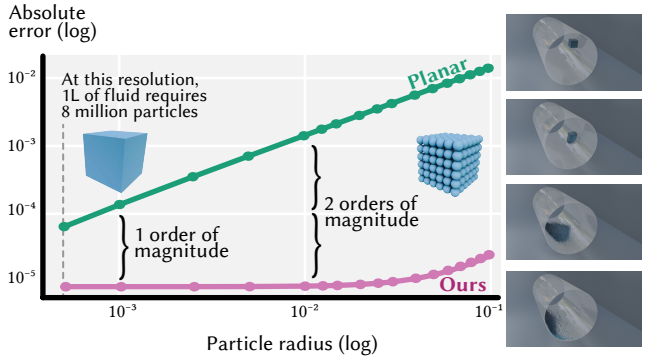


Fig. 9. We consider a litre of fluid interacting with a cylinder of a unit radius. Planar approximations require a very small particle radius to even approach our accuracy to a single order of magnitude.

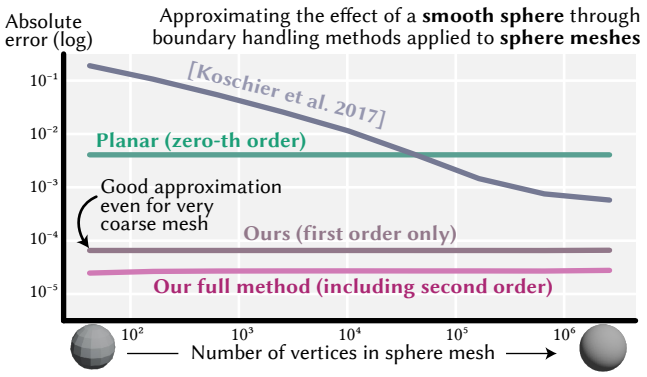


Fig. 10. Meshes are often approximations of smooth underlying geometries. We consider the case in which one wants to simulate the interaction of a fluid with a perfect sphere by using a discrete surface mesh of it, and show that our method provides a more accurate approximation (2-4 orders of magnitude) than prior work even for the coarsest of meshes.

[Skillen et al. 2013]). These pressures are used to correct the preliminary velocities, ensuring incompressibility. Finally, these velocities serve to update the particle positions, ending the simulation step.

Many variations exist on the schematic SPH method presented above; however, most if not all SPH methods rely on estimating each particle's density ρ_i in terms of its immediate neighborhood. When the particle is surrounded by fluid, this can be done with a direct application of Equation 3:

$$\rho_i = \sum_{j \in \mathcal{N}(i)} \frac{m_j}{\rho_j} \rho_j W_{ij} = \sum_{j \in \mathcal{N}(i)} m_j W_{ij}\,, \tag{4}$$

where we abbreviate $W_{ij} = W_h(\mathbf{x}_i - \mathbf{x}_j)$. However, when the particle is near a solid boundary $\partial\Omega$, directly applying Equation 4 would lead to an underestimation of the particle density (see Figure 8, in which the two particles at $\mathbf{x}$ would be assigned radically different density values). We focus this paper on resolving this underestimation in order to provide valid density values at particles near the fluid-solid interface, a task known as *boundary handling*.

3.2 SPH Boundary Handling

A natural way of resolving the above underestimation is to add a *boundary density* correction ρ_{B_i} accounting for the intersection of the particle's neighborhood with the solid:

$$\rho_i = \sum_{j \in \mathcal{N}(i)} m_j W_{ij} + \rho_{B_i}(\mathbf{x}_i) \tag{5}$$

Two main strategies for estimating ρ_{B_i} have been proposed in the SPH literature, which are summarized in Figure 7. *Particle-based methods* [Akinci et al. 2012] rely on a dense sampling of *boundary particles* b_j, which contribute analogously to the fluid ones but whose spatial position is fixed:

$$\rho_B(\mathbf{x}_i) = \sum_{b_j} m_{b_j} \rho_0 W_{i\,b_j}, \tag{6}$$

where the pseudo-mass m_{b_j} may be precomputed or approximated for each particle to account for the interior of the solid, and the kernel W may differ from the fluid kernel. While this is a convenient discretization that aligns with the broader SPH framework, the irregularity of this sampling leads to non-deterministic noise in the simulation that can only be alleviated at a significant computational expense by increasing its density (see Figure 4).

By contrast, *numerical methods* [Bender et al. 2019; Koschier and Bender 2017] consider the exact contribution of the solid to the fluid's density (or the limit of Equation 6 as the number of sampled particles increases), which can be written in integral form as

$$\rho_B(x) = \int_{B_h(\mathbf{x}) \cap \Omega} \rho_0 W_h(||\mathbf{x} - \mathbf{x}'||) d\mathbf{x}' \tag{7}$$

This integral can be computed analytically if the solid boundary $\partial\Omega$ is flat or can be reasonably approximated with planar patches [Fujisawa and Miura 2015; Harada et al. 2007; Winchenbach et al. 2020; Winchenbach and Kolb 2025]. For more complex surfaces, Koschier and Bender [2017] propose a continuous version of this integral that aids with differentiability and numerical stability:

$$\rho_B(x) = \int_{B_h(\mathbf{x})} \gamma(\Phi(x')) W_h(||\mathbf{x} - \mathbf{x}'||) d\mathbf{x}' \tag{8}$$

where Φ is the Signed Distance Function of the boundary, and γ is a continuous piecewise-linear function like

$$\gamma(a) = \begin{cases} \rho_0 \left(1 - \frac{a}{h}\right) & \text{if} \quad a \leq h\,, \\ 0 \quad \text{otherwise.} \end{cases} \tag{9}$$

Koschier and Bender [2017] propose using Gaussian quadrature to estimate the integral in Equation 8, and show that, if computed accurately, it eliminates the irregularity of particle-based algorithms (see Figure 4). An accurate estimation requires a large number of quadrature nodes: for fixed boundaries, the integral can be estimated and cached once before the start of the simulation. However, if the boundaries are changing due to physical forces, prescribed motions or geometric optimization, Equation 8 must be re-estimated at every step of the simulation, becoming a computational bottleneck.

This computational expense is severe enough to make implicit methods impracticable in the simulation of fluids with geometrically

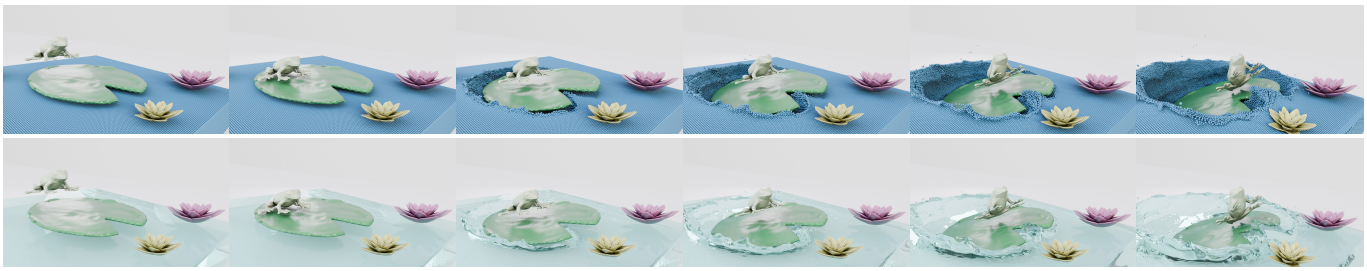


Fig. 11. A deformable frog jumps on a deformable water lily, causing a ripple in the pond. See Supplemental Video for full animation.

changing boundaries. In what follows, we introduce an approximation of Equation 8 that eliminates the need for quadrature estimation and, with it, this computational bottleneck. It will enable us to estimate the value of a fluid particle's boundary density several orders of magnitude more accurately and efficiently, even for moderately complex curved surfaces.

4 Method

Given a fluid particle at a spatial position $\mathbf{x}$, a neighboring solid boundary $\partial\Omega$ and a choice of compactly supported, radially symmetric polynomial kernel W_h, we will compute the particle's boundary density $\rho_B(\mathbf{x})$ using Equation 8. In particular, we will use two separate coordinate transformations to write $\rho_B(\mathbf{x})$ as a polynomial function of the signed distance at $\mathbf{x}$ and the mean and Gaussian curvatures at its closest point on the surface. This polynomial can be easily calculated at runtime using floating-point arithmetic, or tabulated and cached, without the need for numerical integration.

4.1 Boundary density in tubular coordinates

Let d be the signed distance $\Phi(\mathbf{x})$ and $\mathbf{x}^*$ be the closest point to $\mathbf{x}$ on the surface $\partial\Omega$ such that $d = \|\mathbf{x}-\mathbf{x}^*\|$. Let H and K be the mean and Gaussian curvatures of $\partial\Omega$ at $\mathbf{x}^*$, and κ_1, κ_2, $\mathbf{e}_1, \mathbf{e}_2$, be its principal curvature and curvature directions, respectively, such that

$$H = \frac{1}{2}(\kappa_1 + \kappa_2), \quad K = \kappa_1\kappa_2 . \tag{10}$$

In a local neighborhood of $\mathbf{x}^*$, any point on the surface $\mathbf{r}$ can be parametrized as

$$\mathbf{r}(u,v) = \mathbf{x}^* + u\mathbf{e}_1 + v\mathbf{e}_2 + \frac{1}{2}\left(\kappa_1 u^2 + \kappa_2 v^2\right)\mathbf{n}_0 + O\left(\|(u,v)\|^3\right) . \tag{11}$$

Let $\mathbf{n}_0$ be the normal direction of the surface at $\mathbf{x}^*$. Then, a Taylor expansion around $\mathbf{x}^*$ lets us write the normal direction at $\mathbf{r}(u,v)$ as

$$\mathbf{n}(u,v) = \mathbf{n}_0 - u\kappa_1\mathbf{e}_1 - v\kappa_2\mathbf{e}_2 + O\left(\|(u,v)\|^2\right) . \tag{12}$$

Considering small displacements from the surface in this normal direction lets us parametrize all points $\mathbf{x}'$ in a narrow volumetric band around the surface as

$$\mathbf{x}'(u,v,s) = \mathbf{r}(u,v) + s\mathbf{n}(u,v) \tag{13}$$

If $|s|$ is smaller than the surface's local feature size, this mapping is guaranteed to be bijective [Lee 2003]. The parametrized narrow band is known as a *tubular neighborhood* of $\partial\Omega$ and u, v, s are known as its *tubular coordinates*. In this coordinate frame, the boundary integral equation

$$\rho_B(x) = \int_{B_h(\mathbf{x})\cap\Omega} \gamma(\Phi(x'))W_h(||\mathbf{x}-\mathbf{x}'||)d\mathbf{x}' \tag{14}$$

becomes

$$\rho_B(x) = \int_{-h}^{h}\int_{-h}^{h}\int_{-h}^{h} J(s)\gamma(s)W_h(||\mathbf{x}-\mathbf{x}'(u,v,s)||)du\,dv\,ds \tag{15}$$

where $J(s)$ is the Jacobian of the tubular coordinate transformation, which takes the form (see A in the Supplementary Material)

$$J(s) = 1 - 2Hs + Ks^2 . \tag{16}$$

4.2 From 3D integral to 1D integral

Equation 15 allows us to rewrite the boundary density integral in terms involving local geometric properties of the surface. Critically, this new formulation separates the terms in the integral that depend on the normal displacement s (J and γ), and a single term $W_h(||\mathbf{x}-\mathbf{x}'(u,v,s)||)$ storing all the dependency on the displacement along the surface given by u, v:

$$\rho_B(\mathbf{x}) = \int_{-h}^{h} J(s)\gamma(s)\left(\int_{-h}^{h}\int_{-h}^{h} W_h(||\mathbf{x}-\mathbf{x}'(u,v,s)||)du\,dv\right)ds .$$

For later derivations, it will be convenient to also include the tubular Jacobian $J(s)$ in the kernel integral

$$\rho_B(\mathbf{x}) = \int_{-h}^{h} \gamma(s)\left(\int_{-h}^{h}\int_{-h}^{h} J(s)W_h(||\mathbf{x}-\mathbf{x}'(u,v,s)||)du\,dv\right)ds .$$

We will resolve the integral in two steps: first, we will obtain a closed-form polynomial approximation

$$F(s) = \int_{-h}^{h}\int_{-h}^{h} J(s)W_h(||\mathbf{x}-\mathbf{x}'(u,v,s)||)du\,dv , \tag{17}$$

and then solve the one-dimensional polynomial integral

$$\rho_B(\mathbf{x}) = \int_{-h}^{h} \gamma(s)F(s)\,ds . \tag{18}$$

Let us begin with $F(s)$. Given Equation 13, we can write the displacement vector $\mathbf{x}-\mathbf{x}'$ as

$$\mathbf{x}-\mathbf{x}' = \mathbf{x}^* + d\mathbf{n}_0 - \mathbf{r}(u,v) - s\mathbf{n}(u,v) \tag{19}$$

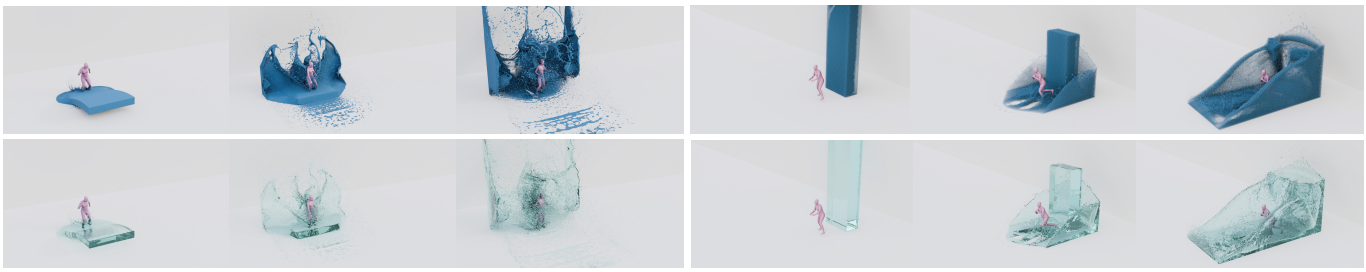


Fig. 12. Two motion capture sequences from [Mahmood et al. 2019], projected on a human body model [Pavlakos et al. 2019], interact with fluids coming from an emitter (left) and a dam break (right). See Supplemental Video for full animation.

which, separating by basis component $\mathbf{e}_1, \mathbf{e}_2, \mathbf{n}_0$ and substituting the expressions in Eqs.11 and 12, leads to

$$\mathbf{x} - \mathbf{x}' = \Delta_0 \mathbf{n}_0 + \Delta_1 \mathbf{e}_1 + \Delta_2 \mathbf{e}_2 \,, \tag{20}$$

where

$$\Delta_0 = d - s - \frac{\kappa_1 u^2 + \kappa_2 v^2}{2} \,, \quad \Delta_1 = -u + s\kappa_1 u \,, \quad \Delta_2 = -v + s\kappa_2 v \,.$$

Since $\mathbf{n}_0$, $\mathbf{e}_1$ and $\mathbf{e}_2$ form an orthonormal basis,

$$\|\mathbf{x} - \mathbf{x}'\|^2 = \Delta_0^2 + \Delta_1^2 + \Delta_2^2 \,. \tag{21}$$

which, expanding the squares, leads to

$$\|\mathbf{x} - \mathbf{x}'\|^2 = u^2 + v^2 + (d-s)^2 - (d+s)(\kappa_1 u^2 + \kappa_2 v^2) + O(h^4).$$

Discarding fourth-order terms, this leads to a new estimate for $F(s)$:

$$\iint_{-h}^{h} J(s) W_h \left(\sqrt{u^2 + v^2 + (d-s)^2 - (s+d)(\kappa_1 u^2 + \kappa_2 v^2)} \right) du\, dv \,.$$

We will simplify this expression further through another transformation, this time to polar coordinates p, θ:

$$u = p \cos\theta \,, \quad v = p \sin\theta \tag{22}$$

such that

$$F(s) \approx 2\pi \int_{-h}^{h} \int_{-h}^{h} p J(s) W_h \left(\sqrt{p^2 + (d-s)^2 + \delta(p,\theta,s)} \right) d\theta\, dp \,.$$

where

$$\delta(p,\theta,s) = -(s+d)(\kappa_1 u^2 + \kappa_2 v^2) \,. \tag{23}$$

Note that although the integrals are written over the box $[-h,h]^2$, the kernel W_h has compact support, so only points satisfying $\|\mathbf{x} - \mathbf{x}'\| \le h$ contribute.

Linearizing the square root and carrying out a Taylor expansion of W_h around $t_0 = \sqrt{p^2 + z^2}$, we can write

$$F(s) \approx 2\pi \iint_{-h}^{h} p J(s) \left(W_h(t_0) + W_h'(t_0)\, \delta(p,\theta,s) + O(h^4) \right) d\theta\, dp \,.$$

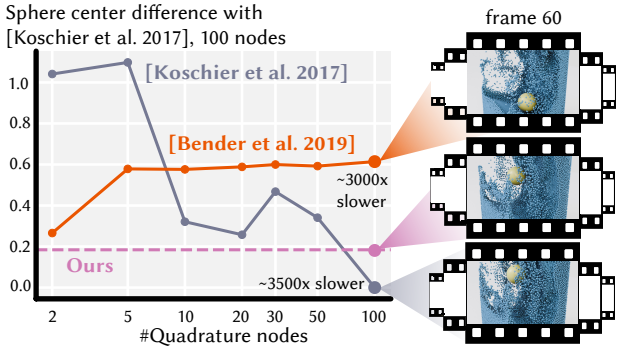


Fig. 13. In this simple simulation, we track the position of a rigid yellow sphere interacting with a block of fluid. Density Maps [Koschier and Bender 2017] introduce a tradeoff between speed and precision, approaching our output only when orders of mangitude slower.

Expanding $J(s)$ to be $J(s) = 1 - 2Hs - Ks^2$, this becomes

$$F(s) \approx 2\pi \iint p \Big(W_h(t_0) + W_h'(t_0)\, \delta(p,\theta,s) - 2HsW_h(t_0) - Ks^2 W_h(t_0) - 2HsW_h'(t_0)\, \delta(p,\theta,s) + O(h^4) \Big) d\theta\, dp \,.$$

We can group these terms by their degree in s:

$$F(s) \approx A_0(s) + A_1(s) + A_2(s) + O(h^4) \tag{24}$$

where

$$A_0(s,H,K) = 2\pi \iint p W_h(t_0)\, d\theta\, dp \,,$$

$$A_1(s,H,K) = 2\pi \iint p \left(-2HsW_h(t_0) + W_h'(t_0)\, \delta(p,\theta,s) \right) d\theta\, dp \,,$$

$$A_2(s,H,K) = -2\pi \iint p \left(Ks^2 W_h(t_0) + 2HsW_h'(t_0)\, \delta(p,\theta,s) \right) d\theta\, dp \,.$$

We refer to these three integrals as the *planar, first and second order momenta* of our tubular density approximation, respectively (see Appendix B in the Supplementary for their precise expression).

4.3 A 1D integral with a closed form solution

The prior derivation of $F(s)$, combined with the general expression for the boundary density in tubular coordinates in Equation 18 and

discarding fourth-order terms, results in

$$\rho_B(\mathbf{x}) \approx \int_{-h}^{h} \gamma(s)\,(A_0(s,H,K) + A_1(s,H,K) + A_2(s,H,K))\,ds\,.$$

Regardless of how arduous the derivation of each A_i may be, they are all polynomials in s of degree at most 2 (see B in the Supplementary Material). Since $\gamma(s)$ is piecewise linear in s, the integrand above is a polynomial of degree at most 3, and therefore can be integrated analytically.

Because the kernel has compact support, the projected kernel moments must be expressed differently depending on whether the integration domain fully lies inside the kernel support or partially intersects its boundary. This as well as the gamma function cutoff induces a number of integration intervals:

$$\rho_B(\mathbf{x}) \approx \int_{I^-_{\text{out}}} P^-_{\text{out}}(s)\,ds \;+\; \int_{I_{\text{in}}} P_{\text{in}}(s)\,ds \;+\; \int_{I^+_{\text{out}}} P^+_{\text{out}}(s)\,ds \quad (25)$$

on each of which the integrand is a cubic polynomial in s

$$P_{\text{in}}(s) = \sum_{k=0}^{3} p^{\text{in}}_k(H,K,d)\,s^k, \qquad P^{\pm}_{\text{out}}(s) = \sum_{k=0}^{3} p^{\text{out},\pm}_k(H,K,d)\,s^k.$$

Since each integrand in Equation 25 is cubic in s, each term can be integrated analytically in closed form. The boundary density thus reduces to a polynomial expression in the signed distance d,

$$\rho_B(\mathbf{x}) \approx \sum_{k=0}^{4} c_k(H,K)\,d^k, \quad (26)$$

where the coefficients c_k can be derived in closed form for an arbitrary kernel and cutoff function (see Appendix C).

5 Results

We implement our algorithm as an additional boundary handling method in a public fork[1] of the popular C++ SPH simulation library SPlisHSPlasH [Bender 2025] . For experiments involving deformables, we make use of the Position Based Dynamics library [Bender et al. 2015] (see D in the Supplementary). We report timings conducted on a Macbook Pro laptop with an M4 Pro chip and 48 GB of memory, and render our results using BlenderToolbox [Liu 2023] and SplashSurf to produce fluid surface meshes [Löschner et al. 2023].

Our closed-form expression for Equation 8 relies on the solid's curvature. We experimented with different discrete curvature estimation methods, like osculating circle radii and vertex angle deficit, but eventually selected LIBIGL's local quadric fitting for its experimental robustness (see Figure 14).

Our quantitative results rely on the direct evaluation of our polynomial integral solution at each particle's position with no grid interpolation and a rest density of 1 g/mL (e.g., Figures 3, 10 and 14). To most seamlessly integrate our method into existing boundary handling implementations, for our medium and large-scale simulation examples, we compute and cache Tube Map values in a grid identical to the one used by Koschier and Bender [2017].

[1]https://gatc.cs.columbia.edu/projects/tubemaps.html

Table 1. Density values calculation and total boundary map (BM) construction cost (including SDF and curvature computation) in seconds, fraction of the simulation timestep spent on BM, and speedup of our method over Density Maps [Koschier and Bender 2017], using the author-provided default of 50^3 quadrature nodes for experiments with deforming solids.

Fig.	Method	Density time (s)	Total BM time (s)	Density speedup	Total BM speedup
12 ←	Density Maps	17.15	17.42	**571×**	**183×**
	Ours	0.03	0.09		
12 →	Density Maps	17.09	17.42	**570×**	**210×**
	Ours	0.03	0.08		
5	Density Maps	1.88	1.89	**188×**	**83×**
	Ours	0.01	0.02		
15 ←	Density Maps	1026	1028	**460×**	**195×**
	Ours	2.23	5.27		
15 ↑	Density Maps	16.09	16.12	**536×**	**195×**
	Ours	0.03	0.06		
11	Density Maps	45.1	44.18	**564×**	**258×**
	Ours	0.08	0.17		
19	Density Maps	15.01	15.15	**500×**	**93×**
	Ours	0.03	0.162		

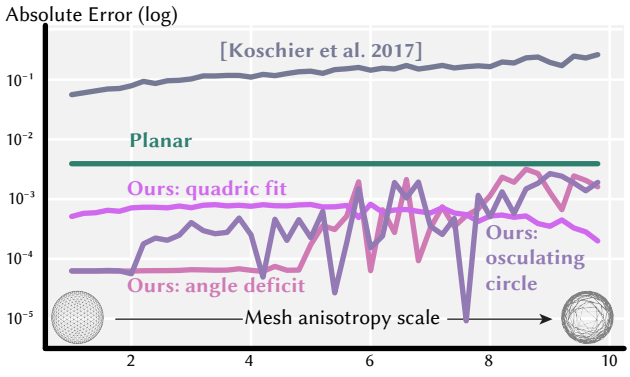


Fig. 14. Despite relying on discrete curvature estimation, our method outperforms planar and numerical strategies even for highly anisotropic meshes.

In all experiments, groundtruths are computed through Monte Carlo integration with a very large number of samples. Comparisons to the works by Koschier and Bender [2017] and Bender et al. [2019] are done using the authors' official implementations. To represent methods that approximate the boundary density using planar approximations in the context of our comparisons (e.g., [Fujisawa and Miura 2015; Winchenbach et al. 2020; Winchenbach and Kolb 2025]), we consider a variant of our method in which the higher-order momenta A_1 and A_2 are discarded, and label it as "Planar" (see E in the Supplementary).

Experiments and Comparisons. Through quantitative evaluations, we demonstrate our algorithm's capabilities as a fast substite for existing costly boundary handling strategies. In Figure 3, we analyze our method's performance on three simple shapes for which signed distances and curvatures are known analytically. It highlights the

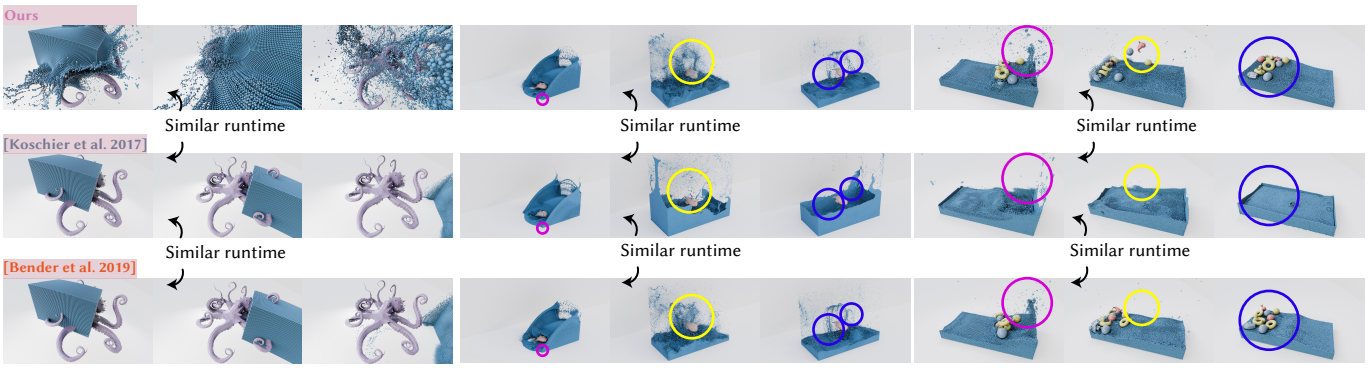


Fig. 15. **Left**: A deformable octopus is hit by the water block from the emitter. If forced to match our runtime by reducing the number of integration nodes, numerical methods Bender et al. [2019]; Koschier and Bender [2017] cause the fluid block not to interact with the object at all. **Center**: A red fish splashes on a water pool created from a dam break simulation. **Right**: A group of rigid objects fall on a deep pool in this example inspired by Koschier and Bender [2017] (see their Fig. 1). The choice of boundary handling method influences the objects' trajectories. See Supplemental Video.

power of our method's geometric approximation: despite geometric simplicity, numerical quadrature methods must rely on an excessive number of quadrature nodes to approach the accuracy that we obtain with a single polynomial evaluation.

Prior work has obtained equally fast boundary density evaluations by relying on planar approximations [Fujisawa and Miura 2015; Winchenbach et al. 2020]. As we show in Figure 9, these approximations are valid when the particle radius is small; however, at medium and large sizes, they pose an undeniable tradeoff between speed and precision. In Figure 6, this lack of accuracy can even be observed qualitatively.

While many simulators use triangle meshes as solid representations, one must bear in mind that these are usually themselves a geometric approximation of a true underlying surface. Our algorithm is designed not to best match the fluid's behaviour when in contact with the piecewise flat surface of the mesh, but its underlying geometry. We make this explicit in Figure 10, where a mesh of a sphere is made progressively finer and different boundary density computation methods are compared against an analytical sphere's groundtruth. Our use of a geometric curvature approximation allows our method to achieve an accuracy that others can only reach with orders of mangitude of additional refinement.

Applications. Once our choice of method has been validated experimentally, we use prototypical examples of fluid simulation scenes to show its nature as a plug-in substitute for prior boundary handling strategies. In Figure 17, we use a sample scene from the SPlisH-SPlasH library [Bender 2025], in which we modify the rotating blade to be curved. In Figure 15 (right), we use rigid bodies to design a swimming pool scene as an *homage* to Figure 1 in the original *Density Maps* publication Koschier and Bender [2017].

The computational benefits of our method will be most noticable in scenes in which the solid changes shape. In these cases, methods like those proposed by Koschier and Bender [2017] and Bender et al. [2019] must recompute their numerical integral at every simulation timestep, a computational bottleneck that makes most moderate-scale simulations impracticable. In Figure 15 (center), a *Splashing*

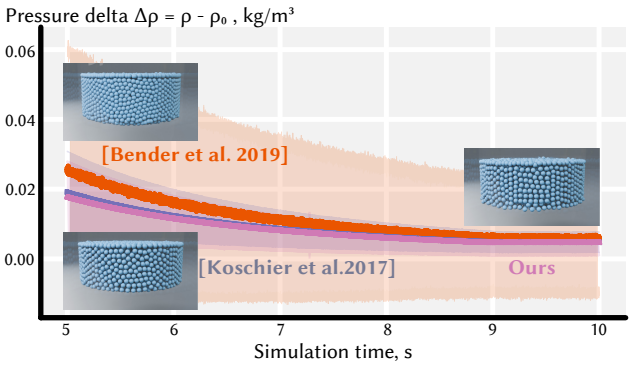


Fig. 16. Pressure profile for a water column at rest near a cylindrical container, with gravity. See Supplemental Video for full animation.

Fish stretches and jumps up and down on a scene with 1.46M particles. In Figure 11, a deforming frog jumps on a pond, pushing a water lily downwards and creating a ripple. In Figure 12, we use the SMPL-X Gender Neutral model [Pavlakos et al. 2019] together with captured motions [Mahmood et al. 2019] to simulate scenes of humans interacting with a complex fluid. In Figure 5 and Figure 1, bouncing and stretching letters are placed between a dam breaking and an invisible wall. In Figure 19, a mountain gets progressively eroded as rain falls on it.

In all these, our method produces visually plausible results while reducing the cost of boundary handling by between two and three orders of magnitude when compared to the germinal implicit boundary handling work by Koschier and Bender [2017] (see Table 1). To further evaluate the impact of this speedup, in Figure 15 we reduce the number of quadrature nodes used by Density Maps and Volume Maps to 8, leading to a runtime similar to our method's. The numerical integration error incurred at such small number of nodes leads to visible differences; e.g., in Figure 15 (left), the block of fluid passes almost entirely through the object without interacting with it. We explore prior work's tradeoff between speed and precision further in Figure 13. Our method also demonstrates low levels of

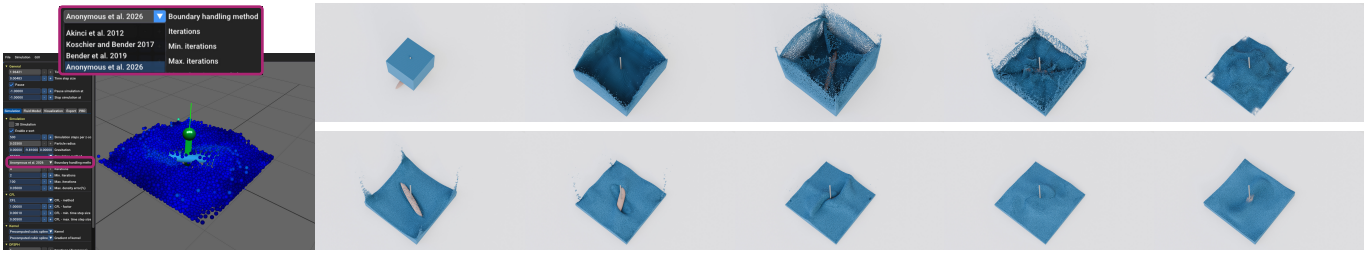


Fig. 17. We implement our algorithm as a novel boundary handling method in a fork of SPlisHSPlasH [Bender 2025], highlighting its role as a drop-in replacement for existing strategies. See Supplemental Video for full animation.

artificial layering or drift near walls, as shown in Figure 16 for a hydrostatic experiment. All full simulations are provided in our **Supplemental Video**.

6 Limitations & Conclusion

We have introduced *Tube Maps*, a curved geometric approximation that enables computation of a fluid particle's boundary density contribution with a single polynomial evaluation. Our formula can be used as a drop-in replacement for prior implicit boundary handling strategies, reducing their cost by several orders of magnitude without reducing their accuracy. This targeted boundary handling strategy is agnostic to the specific SPH algorithm used and can be incorporated into most existing simulation environments.

To improve the accuracy of prior work, which occassionally approximates solid boundaries with planar patches, we consider a curved surface's higher order local geometric properties. This requires estimating the surface's mean and Gaussian curvatures: if the solid is represented as a mesh, this introduces a relation between simulation accuracy and mesh quality , which we explore further in Figure 14). Additionally, this results in inherently smoothing geometric sharp features (see Figure 18). In practice, we omit higher-order momenta when curvature exceeds a prescribed threshold. If the solid is represented as a SDF, it relies on computing differential properties that can be costly in some settings (e.g., neural SDFs [Park et al. 2019]).

Finally, the combination of stability (relative to particle-based methods) and speed (relative to numerical methods) provided by our method will likely be useful in differentiable optimization settings in which the solid boundaries themselves are changing; e.g., topological optimization of fluidic devices. While we have not explored this avenue yet, we hope to have smoothed the path for others to wade into this potentially rich application space of SPH.

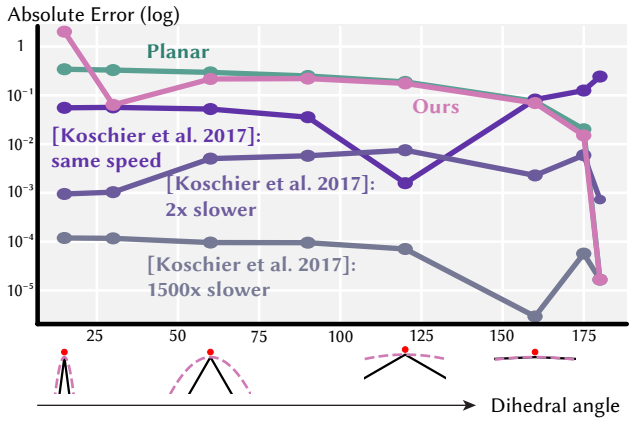


Fig. 18. Our algorithm assumes a smooth solid. When faced with sharp features, it underperforms numerical methods, by an amount proportional to their increase in computational cost.

Fig. 19. A mountain is eroded due to rain falling on it

Acknowledgments

The Geometry and the City lab at Columbia University is supported by generous gifts from nTop, Adobe, Dandy and Braid Technologies, as well as by a sponsored research project from Dreamsports and the Columbia Engineering Interdisciplinary Research Fund. We thank Hyunwoo (Brian) Kim for technical help with prescribed motion sequences; Eitan Grinspun, Changxi Zheng and Christopher Batty for insightful conversations on fluid simulation.

We acknowledge and thank Carlos Cardona for authoring the Blender shader used to render our results, and the authors of the 3D models used throughout this paper. Figures in this work contain the frog [TK3DPrinting 2016], water lily [guppyk 2020], octopus [Banana 2024], fish [C4robotics 2015], seal [Ollender 2024] and ship [ABIG13 2025] meshes.

References

ABIG13. 2025. Inflatable benchy. In *https://www.thingiverse.com/thing:6941422*.

Nadir Akinci, Jens Cornelis, Gizem Akinci, and Matthias Teschner. 2013. Coupling elastic solids with smoothed particle hydrodynamics fluids. *Computer Animation and Virtual Worlds* 24, 3-4 (2013), 195–203.

Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. 2012. Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–8.

Joe Banana. 2024. Octopus. In *https://www.thingiverse.com/thing:6434927*.

Stefan Band, Christoph Gissler, Andreas Peer, and Matthias Teschner. 2018. MLS pressure boundaries for divergence-free and viscous SPH fluids. *Computers & Graphics* 76 (2018), 37–46.

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*. 209–217.

Jan Bender. 2025. SPlisHSPlasH: SPH simulation of fluids and solids. https://splishsplash.physics-simulation.org.

Jan Bender and Dan Koschier. 2015. Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics symposium on computer animation*. 147–155.

Jan Bender, Tassilo Kugelstadt, Marcel Weiler, and Dan Koschier. 2019. Volume maps: An implicit boundary representation for SPH. In *Proceedings of the 12th ACM SIGGRAPH Conference on Motion, Interaction and Games*. 1–10.

Jan Bender, Matthias Müller, and Miles Macklin. 2015. Position-Based Simulation Methods in Computer Graphics.. In *Eurographics (tutorials)*. 8.

C4robotics. 2015. Magikarp. In *https://www.thingiverse.com/thing:1187825*.

PW Cleary, J Ha, M Prakash, and T Nguyen. 2006. 3D SPH flow predictions and validation for high pressure die casting of automotive components. *Applied Mathematical Modelling* 30, 11 (2006), 1406–1427.

Zhiyang Dou, Chen Peng, Xinyu Lu, Xiaohan Ye, Lixing Fang, Yuan Liu, Wenping Wang, Chuang Gan, Lingjie Liu, and Taku Komura. 2025. CFC: Simulating Character-Fluid Coupling using a Two-Level World Model. *ACM Transactions on Graphics (TOG)* 44, 6 (2025), 1–17.

Florian Fleissner, Alexandra Lehnart, and Peter Eberhard. 2010. Dynamic simulation of sloshing fluid and granular cargo in transport vehicles. *Vehicle system dynamics* 48, 1 (2010), 3–15.

Makoto Fujisawa and Kenjiro T Miura. 2015. An efficient boundary handling with a modified density calculation for SPH. In *Computer graphics forum*, Vol. 34. Wiley Online Library, 155–162.

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.

Christoph Gissler, Andreas Peer, Stefan Band, Jan Bender, and Matthias Teschner. 2019. Interlinked SPH pressure solvers for strong fluid-rigid coupling. *ACM Transactions on Graphics (TOG)* 38, 1 (2019), 1–13.

guppyk. 2020. Lotus Flower / Water Lily with Leaves. In *https://www.thingiverse.com/thing:4364395*.

Ricardo Gutfraind and Stuart B Savage. 1998. Flow of fractured ice through wedge-shaped channels: smoothed particle hydrodynamics and discrete-element simulations. *Mechanics of materials* 29, 1 (1998), 1–17.

Takahiro Harada, Seiichi Koshizuka, and Yoichiro Kawaguchi. 2007. Smoothed particle hydrodynamics in complex shapes. In *Proceedings of the 23rd Spring Conference on Computer Graphics*. 191–197.

Markus Ihmsen, Jens Orthmann, Barbara Solenthaler, Andreas Kolb, and Matthias Teschner. 2014. SPH fluids in computer graphics. (2014).

Dan Koschier et al. [n. d.]. *Discregrid Library*. https://github.com/InteractiveComputerGraphics/Discregrid

Dan Koschier and Jan Bender. 2017. Density maps for improved SPH boundary handling. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 1–10.

Dan Koschier, Jan Bender, Barbara Solenthaler, and Matthias Teschner. 2020. Smoothed particle hydrodynamics techniques for the physics based simulation of fluids and solids. *arXiv preprint arXiv:2009.06944* (2020).

Dan Koschier, Jan Bender, Barbara Solenthaler, and Matthias Teschner. 2022. A survey on SPH methods in computer graphics. In *Computer graphics forum*, Vol. 41. Wiley Online Library, 737–760.

BE Launder and DB Spalding. 1972. Mathematical models of turbulence Academic Press, London and N.

David Le Touzé, A Marsh, G Oger, Pierre-Michel Guilcher, C Khaddaj-Mallat, B Alessandrini, and P Ferrant. 2010. SPH simulation of green water and ship flooding scenarios. *Journal of Hydrodynamics* 22, 1 (2010), 231–236.

John M Lee. 2003. Smooth manifolds. In *Introduction to smooth manifolds*. Springer, 1–29.

Zhehao Li, Qingyu Xu, Xiaohan Ye, Bo Ren, and Ligang Liu. 2023. DiffFR: Differentiable SPH-based Fluid-Rigid Coupling for Rigid Body Control. *ACM Trans. Graph.* 42, 6 (Dec 2023), 17 pages. https://doi.org/10.1145/3618318

Hsueh-Ti Derek Liu. 2023. BlenderToolbox. https://github.com/HTDerekLiu/BlenderToolbox.

Fabian Löschner, Timna Böttcher, Stefan Rhys Jeske, and Jan Bender. 2023. Weighted Laplacian Smoothing for Surface Reconstruction of Particle-based Fluids. In *Vision, Modeling, and Visualization*. The Eurographics Association. doi:10.2312/vmv.20231245

Shang Ma, Xiaoying Nie, Gang Yang, and Chunqing Zhou. 2025. A robust and efficient model for the interaction of fluids with deformable solids. *The Visual Computer* (2025), 1–14.

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. 2019. AMASS: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5442–5451.

S Marrone, A Colagrossi, M Antuono, C Lugni, and MP Tulin. 2011. A 2D+ t SPH model to study the breaking wave pattern generated by fast ships. *Journal of Fluids and Structures* 27, 8 (2011), 1199–1215.

Petra Ollender. 2024. Seal on a floater. In *https://makerworld.com/ru/models/409987-seal-on-a-floater-bathtub-toy*.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. 2019. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 165–174.

Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. 2019. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10975–10985.

Jacek Pozorski and Michał Olejnik. 2024. Smoothed particle hydrodynamics modelling of multiphase flows: an overview. *Acta Mechanica* 235, 4 (2024), 1685–1714.

Martin Siemann and Paul Groenenboom. 2014. Modeling and validation of guided ditching tests using a coupled SPH-FE approach. In *Proceedings of 9th SPHERIC international workshop*. 260–268.

Alex Skillen, Steven Lind, Peter K. Stansby, and Benedict D. Rogers. 2013. Incompressible smoothed particle hydrodynamics (SPH) with reduced temporal noise and generalised Fickian smoothing applied to body–water slam and efficient wave–body interaction. *Computer Methods in Applied Mechanics and Engineering* 265 (2013), 163–173. doi:10.1016/j.cma.2013.05.017

Barbara Solenthaler and Renato Pajarola. 2009. Predictive-corrective incompressible SPH. In *ACM SIGGRApH 2009 papers*. 1–6.

TK3DPrinting. 2016. Low Poly Frog. In *https://www.thingiverse.com/thing:1762864*.

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, Rustam Akhunov, and Andreas Kolb. 2020. Semi-analytic boundary handling below particle resolution for smoothed particle hydrodynamics. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–17.

Rene Winchenbach and Andreas Kolb. 2025. Solving Boundary Handling Analytically in Two Dimensions for Smoothed Particle Hydrodynamics. *arXiv preprint arXiv:2507.21686* (2025).

Rene Winchenbach and Nils Thuerey. 2025. diffSPH: Differentiable Smoothed Particle Hydrodynamics for Adjoint Optimization and Machine Learning. *arXiv preprint arXiv:2507.21684* (2025).

Tube Maps: Fast SPH Boundary Handling in Tubular Coordinates (Supplementary Material)

DARIA NOGINA, Columbia University, USA
SILVIA SELLÁN, Columbia University, USA

ACM Reference Format:
Daria Nogina and Silvia Sellán. 2026. Tube Maps: Fast SPH Boundary Handling in Tubular Coordinates (Supplementary Material). In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '26), July 19–23, 2026, Los Angeles, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3799902.3811118

Supplementary Material

A Jacobian for the offset surface

The area element of the base surface is

$$dA = \|r_u \times r_v\| \, du \, dv. \tag{1}$$

Consider the parallel surface at signed distance s,

$$r_s(u,v) = r(u,v) + s\, n(u,v).$$

Its tangent vectors are

$$t_u = r_u + s n_u, \qquad t_v = r_v + s n_v. \tag{2}$$

Using the Weingarten equations,

$$n_u = -S r_u, \qquad n_v = -S r_v,$$

we obtain

$$t_u = (I - sS) r_u, \qquad t_v = (I - sS) r_v. \tag{3}$$

The area element of the offset surface is therefore

$$\begin{aligned} dA_{\text{offset}} &= \|t_u \times t_v\| \, du \, dv \\ &= \|(I - sS) r_u \times (I - sS) r_v\| \, du \, dv \\ &= |\det(I - sS)| \, \|r_u \times r_v\| \, du \, dv. \end{aligned} \tag{4}$$

Hence the area scale factor is

$$J(s) = \det(I - sS).$$

In the principal frame $\{e_1, e_2, n\}$, the shape operator has the form

$$S = \operatorname{diag}(\kappa_1, \kappa_2, 0),$$

so that

$$I - sS = \operatorname{diag}(1 - \kappa_1 s,\ 1 - \kappa_2 s,\ 1),$$

and therefore

$$\det(I - sS) = (1 - \kappa_1 s)(1 - \kappa_2 s) = 1 - (\kappa_1 + \kappa_2) s + \kappa_1 \kappa_2 s^2.$$

Using the conventions

$$H = \tfrac{1}{2}(\kappa_1 + \kappa_2), \qquad K = \kappa_1 \kappa_2,$$

this becomes

$$J(s) = 1 - 2Hs + Ks^2.$$

Consequently, the area element of the offset surface satisfies

$$dA_{\text{offset}} = (1 - 2Hs + Ks^2)\, dA. \tag{5}$$

B Derivation of the momenta $A_i(s, H, K)$

B.1 A_0

The zeroth-order term corresponds to retaining only the planar distance $t_0 = \sqrt{p^2 + z^2}$ in the kernel argument and using the zeroth-order Jacobian $J(s) \approx 1$. Writing the tangential coordinates in polar form $u = p\cos\theta$, $v = p\sin\theta$ so that $du\,dv = p\,dp\,d\theta$, we obtain

$$A_0(s,H,K) = \iint W_h\big(\sqrt{u^2 + v^2 + z^2}\big)\, du\, dv = \int_0^{2\pi} \int_0^{\infty} p\, W_h(t_0)\, dp\, d\theta. \tag{6}$$

Since W_h has compact support of radius h, the integrand is nonzero only when $t_0 \le h$, i.e. when $p^2 + z^2 \le h^2$. Thus the p-integration domain is $p \in [0, \sqrt{h^2 - z^2}]$ for $|z| \le h$, and empty otherwise. Carrying out the angular integral gives

$$A_0(s,H,K) = 2\pi \int_0^{\sqrt{h^2 - z^2}} p\, W_h(t_0)\, dp, \qquad |z| \le h, \tag{7}$$

and $A_0(s,H,K) = 0$ for $|z| > h$.

B.2 A_1

The first-order term collects all contributions linear in curvature. These arise from the linear term in the Jacobian expansion $J(s) = 1 - 2Hs + O(\kappa^2)$ and the first-order correction to the kernel argument through $W_h'(t_0)\, \delta t$.

Recall the full distance expansion in Eq. 21. With $p^2 = u^2 + v^2$ and $z = d - s$, the omitted part of $\|x - x'\|^2$ is $\Delta = -(z + 2s)(\kappa_1 u^2 + \kappa_2 v^2)$. Let $t = \sqrt{t_0^2 + \Delta}$ be the "true" distance. Using $u = p\cos\theta$, $v = p\sin\theta$, for fixed s (hence fixed z), points with radius p lie on a ring of length $2\pi p$ in the (u,v)-plane we have

$$\kappa_1 u^2 + \kappa_2 v^2 = p^2(\kappa_1 \cos^2\theta + \kappa_2 \sin^2\theta),$$

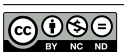
$$\int_0^{2\pi} (\kappa_1 u^2 + \kappa_2 v^2)\, d\theta = p^2\big(\kappa_1 \pi + \kappa_2 \pi\big) = \pi p^2 (\kappa_1 + \kappa_2) = 2\pi p^2 H,$$

hence the angular average satisfies $\langle \kappa_1 u^2 + \kappa_2 v^2 \rangle_\theta = p^2 H$ and therefore

$$\langle \Delta \rangle_\theta = -(z + 2s)\, H\, p^2, \qquad \langle \delta t \rangle_\theta \approx -(z + 2s)\, H\, \frac{p^2}{2t_0}. \tag{8}$$

, where $\delta t := t - t_0 \approx \frac{\Delta}{2t_0}$.

Authors' Contact Information: Daria Nogina, Columbia University, USA, dsn2136@columbia.edu; Silvia Sellán, Columbia University, USA, silviasellan@cs.columbia.edu.



SIGGRAPH Conference Papers '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2554-8/2026/07
https://doi.org/10.1145/3799902.3811118

The first-order distance contribution to $F(s)$ is obtained by integrating $W_h'(t_0)\langle\delta t\rangle_\theta$ over the ring measure $p\,dp\,d\theta$:

$$A_1^{\text{dist}}(s,H,K) := \int_0^{2\pi}\int_0^{\infty} p\,W_h'(t_0)\,\langle\delta t\rangle_\theta\,dp\,d\theta \tag{9}$$

$$= -(z+2s)\,H\,(2\pi)\int_0^{\infty}\frac{p^2}{2t_0}\,W_h'(t_0)\,p\,dp. \tag{10}$$

Using $t_0 = \sqrt{p^2+z^2}$, we have $dt_0/dp = p/t_0$, hence $p\,dp = t_0\,dt_0$ and $p^2 = t_0^2 - z^2$. Restricting to the kernel support $t_0 \le h$ yields

$$A_1^{\text{dist}}(s,H,K) = -(z+2s)\,H\,(2\pi)\int_{|z|}^{h}\frac{t^2-z^2}{2}\,W_h'(t)\,dt. \tag{11}$$

The first-order Jacobian contribution is simply $A_1^J(s,H,K) := (-2Hs)\,A_0(s,H,K)$. Combining both gives the full moment

$$A_1(s,H,K) = -2Hs\,A_0(s,H,K) - (z+2s)\,H\,(2\pi)\int_{|z|}^{h}\frac{t^2-z^2}{2}\,W_h'(t)\,dt.$$

B.3 A_2

The second-order term collects all contributions quadratic in curvature. These arise from the quadratic term in the Jacobian expansion Ks^2, second-order terms in the kernel-distance expansion, and mixed Jacobian–distance terms.

As in the first-order case, the remaining work is to reduce ring integrals involving higher powers of the tangential radius p to one-dimensional radial integrals in $t = \sqrt{p^2+z^2}$. The identities

$$t^2 = p^2 + z^2, \qquad p\,dp = t\,dt, \qquad p^2 = t^2 - z^2$$

will be used repeatedly, with limits $t \in [|z|, h]$ imposed by compact support.

A useful t^3 radial moment. Second-order distance terms produce ring integrals containing $(t^2-z^2)tW(t)$. We introduce the t^3 radial moment

$$M_3(z) := 2\pi\int_{|z|}^{h} t^3\,W_h(t)\,dt. \tag{12}$$

Moreover, rewriting $(t^2-z^2)t = t^3 - z^2 t$ gives

$$2\pi\int_{|z|}^{h}(t^2-z^2)\,t\,W_h(t)\,dt = 2\pi\int_{|z|}^{h}\big(t^3 - z^2t\big)W_h(t)\,dt \tag{13}$$

$$= M_3(z) - z^2\Big(2\pi\int_{|z|}^{h} t\,W_h(t)\,dt\Big). \tag{14}$$

The bracketed term is the radial form of the planar slice:

$$A_0(z) := 2\pi\int_{|z|}^{h} t\,W_h(t)\,dt,$$

so the ring integral can be written as $M_3(z) - z^2A_0(z)$.

A combined second-order radial moment. The particular combination that appears after angular averaging of the second-order curvature terms can be written as

$$M_2(z) := 4z^2\,A_0(z) - 4M_3(z), \tag{15}$$

which combines the (H^2, K) curvature contributions into a single one-dimensional function of z.

Final form of $A_2(s,H,K)$. The quadratic Jacobian contribution is

$$A_2^J(s,H,K) := Ks^2\,A_0(s,H,K). \tag{16}$$

After angular integration, the remaining second-order distance and mixed terms reduce to a combination of the same $W'(t)$ radial integral that appeared in A_1 and the moment $M_2(z)$:

$$A_2(s,H,K) = Ks^2\,A_0(s,H,K) + \tag{17}$$

$$s^2(2H^2-K)\,(2\pi)\int_{|z|}^{h}\frac{t^2-z^2}{2}\,W_h'(t)\,dt + \tag{18}$$

$$\Big(\frac{3}{16}H^2 - \frac{1}{16}K\Big)M_2(z), \qquad z = d - s. \tag{19}$$

C Second-order tubular integral

Using the second-order truncation derived in the main text, the tubular boundary density can be written as

$$\rho_B(x) \approx \int_{s_{\min}}^{s_{\max}}\gamma(s)\Big[\mathcal{A}(s;d,H,K)\,A_0(d-s) + \mathcal{B}(H,K)\,A_2(d-s)\Big]\,ds, \tag{20}$$

where $d = \Phi(x)$ is the signed distance and $z = d - s$. Here $A_0(\cdot)$ is the projected slice moment, and $A_2(\cdot)$ is the combined second-order radial moment (Appendix B). Here it's convenient to introduce a prefactor multiplying A_0 as a quadratic polynomial in s:

$$\mathcal{A}(s;d,H,K) = 1 + \frac{H}{2}d - \frac{H}{2}s + \Big(2K + \frac{H^2}{2}\Big)s^2, \tag{21}$$

and the coefficient multiplying A_2 as

$$\mathcal{B}(H,K) = \lambda_B\Big(\frac{3}{16}H^2 - \frac{1}{16}K\Big). \tag{22}$$

We use the linear extension function as in [Koschier and Bender 2017]

$$\gamma(s) = 1 - \frac{\text{factor}}{h}s, \qquad \gamma(s) = 0 \text{ for } s > s_\gamma, \qquad s_\gamma = \frac{h}{\text{factor}}.$$

Kernel support requires $|d-s| \le h$. Therefore,

$$s \in [s_{\min}, s_{\max}], \qquad s_{\min} = \max(-h, d-h), \qquad s_{\max} = \min(h, d+h, s_\gamma). \tag{23}$$

C.1 Shift to the symmetric variable $t = s - d$

To eliminate the absolute value in the projected moments, we shift

$$t := s - d, \qquad s = d + t, \qquad z = d - s = -t, \qquad |z| = |t|.$$

The integration bounds become

$$t \in [t_{\min}, t_{\max}], \qquad t_{\min} = s_{\min} - d, \qquad t_{\max} = s_{\max} - d.$$

In particular, using the base support interval $[-h, h]$ and the gamma cutoff,

$$t_{\min} = -h, \qquad t_{\max} = \min\big(h,\, s_\gamma - d\big). \tag{24}$$

With $s = d + t$,

$$\gamma(d+t) = 1 - \frac{\text{factor}}{h}(d+t) = g_0 + g_1 t, \tag{25}$$

$$g_0 = 1 - \frac{\text{factor}}{h}d, \quad g_1 = -\frac{\text{factor}}{h}. \tag{26}$$

Expanding $\mathcal{A}(d+t)$ gives a quadratic in t:

$$\mathcal{A}(d+t) = \widetilde{A}_0 + \widetilde{A}_1 t + \widetilde{A}_2 t^2, \tag{27}$$

where, for $\mathcal{A}(s) = c_0 + c_1 s + c_2 s^2$ with

$$c_0 = 1 + \frac{H}{2}d, \qquad c_1 = -\frac{H}{2}, \qquad c_2 = 2K + \frac{H^2}{2},$$

we have

$$\widetilde{A}_0 = c_0 + c_1 d + c_2 d^2, \qquad \widetilde{A}_1 = c_1 + 2c_2 d, \qquad \widetilde{A}_2 = c_2. \tag{28}$$

Multiplying (25) and (27) yields a cubic polynomial in t:

$$(g_0 + g_1 t)(\widetilde{A}_0 + \widetilde{A}_1 t + \widetilde{A}_2 t^2) = E_0 + E_1 t + E_2 t^2 + E_3 t^3, \tag{29}$$

with coefficients

$$E_0 = g_0\widetilde{A}_0, \quad E_1 = g_0\widetilde{A}_1 + g_1\widetilde{A}_0, \quad E_2 = g_0\widetilde{A}_2 + g_1\widetilde{A}_1, \quad E_3 = g_1\widetilde{A}_2.$$

Substituting into (20) gives

$$\rho_B(x) \approx \int_{t_{\min}}^{t_{\max}} \left(E_0 + E_1 t + E_2 t^2 + E_3 t^3\right) A_0(|t|)\, dt + \tag{30}$$

$$\mathcal{B}(H,K) \int_{t_{\min}}^{t_{\max}} (g_0 + g_1 t)\, A_2(|t|)\, dt. \tag{31}$$

C.2 Piecewise polynomial projected moments for the cubic spline kernel

Let

$$q := \frac{|t|}{h} \in [0,1]. \tag{32}$$

For the cubic spline kernel used as the default in the SPlisHSPlasH library [Bender 2025], the projected slice moment $A_0(|t|)$ admits a piecewise degree-5 polynomial representation in q:

$$A_0(|t|) = \frac{16}{h} \begin{cases} P_{\text{in}}(q), & 0 \le q \le \frac{1}{2}, \\ P_{\text{out}}(q), & \frac{1}{2} < q \le 1, \end{cases} \tag{33}$$

where (ascending powers)

$$P_{\text{in}}(q) = 0.0875 - 0.5\, q^2 + 1.5\, q^4 - 1.2\, q^5, \tag{34}$$

$$P_{\text{out}}(q) = 0.1 - 1.0\, q^2 + 2.0\, q^3 - 1.5\, q^4 + 0.4\, q^5. \tag{35}$$

Likewise, the combined second-order radial moment A_2 is represented as a piecewise degree-7 polynomial in q with an overall scaling:

$$A_2(|t|) = \frac{h}{35} \begin{cases} P^B_{\text{in}}(q), & 0 \le q \le \frac{1}{2}, \\ P^B_{\text{out}}(q), & \frac{1}{2} < q \le 1, \end{cases} \tag{36}$$

with (ascending powers)

$$P^B_{\text{in}}(q) = -31 + 196\, q^2 - 560\, q^4 + 1120\, q^6 - 768\, q^7, \tag{37}$$

$$P^B_{\text{out}}(q) = -32 + 224\, q^2 - 1120\, q^4 + 1792\, q^5 - 1120\, q^6 + 256\, q^7. \tag{38}$$

C.3 Knot segmentation

Because (33)–(36) change form at $q = \frac{1}{2}$ and depend on $|t|$, we split the integration range into the canonical segments

$$[-h, -\tfrac{1}{2}h], \quad [-\tfrac{1}{2}h, 0], \quad [0, \tfrac{1}{2}h], \quad [\tfrac{1}{2}h, h], \tag{39}$$

and intersect each with $[t_{\min}, t_{\max}]$.

On any fixed segment, the sign $\sigma = \text{sign}(t) \in \{-1, +1\}$ is constant. Hence we can replace

$$q^k = \left(\frac{|t|}{h}\right)^k = \frac{\sigma^k}{h^k}\, t^k, \tag{40}$$

i.e. odd powers acquire a sign flip on the negative side.

Therefore, on each segment both $A_0(|t|)$ and $A_2(|t|)$ become ordinary polynomials in t.

C.4 Segment primitives

On a given segment, write

$$A_0(|t|) = \sum_{i=0}^{5} a_i\, t^i \qquad \text{(degree 5 in } t\text{)}. \tag{41}$$

Multiplying by the cubic factor from (29) gives a degree-8 polynomial integrand. Rather than expanding all coefficients explicitly, we integrate term-by-term:

$$\int a_i t^i (E_0 + E_1 t + E_2 t^2 + E_3 t^3)\, dt = \tag{42}$$

$$a_i \left(\frac{E_0}{i+1} t^{i+1} + \frac{E_1}{i+2} t^{i+2} + \frac{E_2}{i+3} t^{i+3} + \frac{E_3}{i+4} t^{i+4}\right), \tag{43}$$

summed over $i = 0, \ldots, 5$. Evaluating the resulting primitive at segment endpoints produces the exact contribution on that segment.

Similarly, on the same segment write

$$A_2(|t|) = \sum_{j=0}^{7} b_j\, t^j \qquad \text{(degree 7 in } t\text{)}. \tag{44}$$

Then $(g_0 + g_1 t)A_2(|t|)$ is degree 8, and we again integrate term-by-term:

$$\int b_j t^j (g_0 + g_1 t)\, dt = b_j \left(\frac{g_0}{j+1} t^{j+1} + \frac{g_1}{j+2} t^{j+2}\right), \tag{45}$$

summed over $j = 0, \ldots, 7$. The curvature prefactor $\mathcal{B}(H,K)$ is applied afterwards.

C.5 Final form

Let the four canonical segments (39) be intersected with $[t_{\min}, t_{\max}]$, yielding subintervals $[a_\ell, b_\ell]$ for $\ell = 1, \ldots, 4$. On each segment we choose the appropriate in/out polynomials for A_0 and A_2, apply the sign rule (40) to convert q-polynomials into t-polynomials, and evaluate the exact primitives from (42) and (45). Summing the segment contributions yields

$$\rho_B(x) \approx \sum_{\ell=1}^{4} \Bigg[\int_{a_\ell}^{b_\ell} \left(E_0 + E_1 t + E_2 t^2 + E_3 t^3\right) A_0(|t|)\, dt \tag{46}$$

$$+ \mathcal{B}(H,K) \int_{a_\ell}^{b_\ell} (g_0 + g_1 t)\, A_2(|t|)\, dt \Bigg]. \tag{47}$$

where each integral is evaluated analytically as described above.

D Effective volume computation for PBD coupling

For position-based dynamics (PBD) of deformable solids, we ultimately need forces on solid vertices. These forces originate from fluid-boundary interactions in the SPH step and must be redistributed to the vertices.

D.1 Boundary particles (Akinci)

In the particle-based method of [Akinci et al. 2012] each boundary particle has an effective volume and receives a force from the fluid that does not depend on the solid density. To distribute forces to the solid, it is convenient to accumulate *force density* at boundary particles:

$$f_{b_i} = \frac{F_{b_i \leftarrow f_i}}{V_{\text{eff}}}, \tag{48}$$

where $F_{b_i \leftarrow f_i}$ is the force from fluid particle f_i to boundary particle b_i and V_{eff} is the effective boundary volume,

$$V_{\text{eff}} = \frac{1}{\sum_k W_{ik}}$$

as in [Akinci et al. 2012].

Force densities f_{b_i} are then transferred to solid vertices using barycentric weights on the solid surface. For the solid, actual vertex forces are recovered by multiplying by the vertex volume, i.e. vertex mass divided by solid density.

D.2 Implicit-boundary methods

For implicit-boundary approaches (density maps, volume maps, and our method), we likewise accumulate forces $F_{b \leftarrow f_i}$ that the fluid applies to the boundary region near each fluid particle f_i. The most intuitive strategy is to project these forces to the closest surface point and distribute them to nearby vertices using barycentric weights, analogous to the particle-based approach.

The key question is: what is the correct effective volume V_{eff} associated with this projected point? We define a fictitious boundary particle b_i located at the closest point on the surface, with pseudo-mass $\bar{m}_{\text{eff}}$ (or equivalently volume V_{eff}), such that it reproduces the same boundary density contribution as the continuous density map. That is,

$$\rho_B(x) = \int_{N(x)} \gamma(\Phi(x'))W(\|x - x'\|, h)\, dx' = \bar{m}_{\text{eff}}\, W(\|x_{f_i} - x_{b_i}\|, h). \tag{49}$$

By definition,

$$\bar{m}_{\text{eff}} = \frac{\rho_B(x)}{W(\|x_{f_i} - x_{b_i}\|, h)}. \tag{50}$$

In [Akinci et al. 2012], the fictitious mass is $\bar{m}_{\text{eff}} = V_{\text{eff}}\, \rho_0$, where ρ_0 is the fluid rest density. Thus

$$V_{\text{eff}} = \frac{\rho_B(x)}{\rho_0\, W(\|x_{f_i} - x_{b_i}\|, h)}. \tag{51}$$

For volume-map approach, we can likewise use the estimated boundary volume fraction $V_B(x)$ as an effective volume.

E Details on comparison with linear methods

While both Koschier and Bender [2017] and Winchenbach et al . [2020] approximate the integral in Eq. 8, Winchenbach et al . [2020] use a constant γ while Koschier and Bender [2017] use a linear one. For a fair numerical comparison with Koschier and Bender [2017] and ours, our planar baseline uses the same linear γ, but is otherwise identical to the method proposed by Winchenbach et al . [2020].