Semi-Analytic Boundary Handling Below Particle Resolution for Smoothed Particle Hydrodynamics

RENE WINCHENBACH, University of Siegen
RUSTAM AKHUNOV, University of Siegen
ANDREAS KOLB, University of Siegen

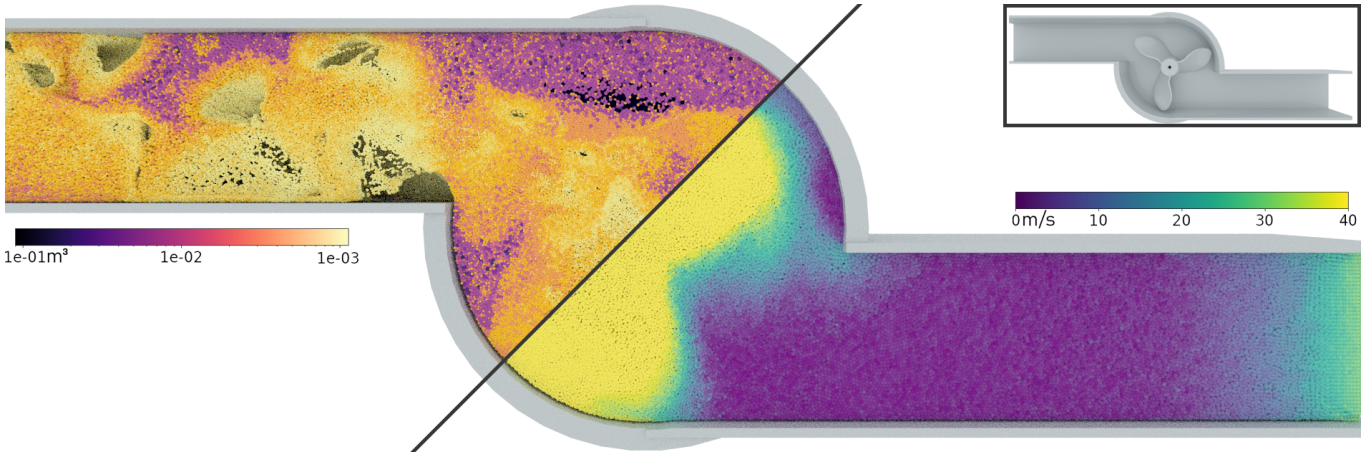


Fig. 1. Our novel semi-analytical boundary handling method enables fluid-rigid interactions even under difficult conditions and can be directly combined with any spatially adaptive simulation technique. This figure shows the simulation of a collision of an inlet flow from the right with a counterclockwise rotating propeller, calculated with up to 1.8 million particles and an adaptive volume ratio of 100 : 1. The boundary configuration is shown in the top right corner, the left and the right part of the main figure visualizes particle volume and particle velocity, respectively.

In this paper, we present a novel semi-analytical boundary handling method for spatially adaptive and divergence-free smoothed particle hydrodynamics (SPH) simulations, including two-way coupling. Our method is consistent under varying particle resolutions and allows for the treatment of boundary features below the particle resolution. We achieve this by first introducing an analytic solution to the interaction of SPH particles with planar boundaries, in 2D and 3D, which we extend to arbitrary boundary geometries using signed distance fields (SDF) to construct locally planar boundaries. Using this boundary-integral-based approach, we can directly evaluate boundary contributions, for any quantity, allowing an easy integration into state of the art simulation methods. Overall, our method improves interactions with small boundary features, readily handles spatially adaptive fluids, preserves particle-boundary interactions across varying resolutions, can directly be implemented in existing SPH methods, and, for non-adaptive simulations, provides a reduction in memory consumption as well as an up to 2× speedup relative to current particle-based boundary handling approaches.

CCS Concepts: • **Computing methodologies** → **Physical simulation**; *Multiscale systems*; Massively parallel and high-performance simulations; • **Mathematics of computing** → Integral equations.

Author's addresses: Rene Winchenbach (rene.winchenbach@uni-siegen.de), Rustam Akhunov (rustam.akhunov@uni-siegen.de) and Andreas Kolb (andreas.kolb@uni-siegen.de), Universität Siegen, Hölderlinstraße 3, 57072 Siegen, Germany.

Additional Key Words and Phrases: SPH, spatial adaptivity, physical simulation, two-way coupling, boundary handling, semi-analytical methods

ACM Reference Format:
Rene Winchenbach, Rustam Akhunov, and Andreas Kolb. 2020. Semi-Analytic Boundary Handling Below Particle Resolution for Smoothed Particle Hydrodynamics. *ACM Trans. Graph.* 39, 6, Article 173 (December 2020), 17 pages. https://doi.org/10.1145/3414685.3417829

1 INTRODUCTION

In modern computer animation the physically accurate simulation of high quality free surface liquid systems is becoming ever more important, but uniform resolution increases are strongly limited by available computational resources. However, not all areas of a simulation are equally important for the outcome, i.e., for computer animation the surface detail is more important than the fluid's bulk, and as such methods have been developed that focus the computational resources in regions of interest, which allow for far greater surface detail than would be possible with a uniform resolution at the same computational requirements. Recent work has enabled incompressible spatially adaptive simulations with large refinement ratios for Smoothed Particle Hydrodynamics (SPH) [Winchenbach et al. 2017]. So far, the robust handling of arbitrary boundaries is an unsolved challenge in adaptive SPH, i.e., methods are required that provide consistent interactions with arbitrary boundaries across widely varying resolution scales. Additionally, representing small

boundary features requires significantly higher fluid resolutions, relative to boundary feature sizes.

In SPH boundaries are commonly described using a particle-based representation [Akinci et al. 2012], which places boundary particles on the surface of an object, or using some direct representation of the boundary [Bender et al. 2019], i.e., by pre-calculating boundary integrals based on, for instance, grid-based integration schemes. Particle-based methods suffer from a dependence on the sampling quality and while there are methods that can change the particle distribution on-the-fly, there are no methods that can change the resolution of a boundary sampling on-the-fly. Direct boundary representation methods suffer from a dependence on the sampling resolution. Koschier and Bender [2017] precompute a boundary-integral on a regular grid, which is only correct for one resolution, whilst Fujisawa and Miura [2015] utilize empirical methods, to enhance a semi-analytical solution, which are not transferrable across varying particle resolutions. In Computational Fluid Dynamics (CFD) boundaries are commonly described using wall-renormalization approaches; see Feldman and Bonet [2007]. These can be realized in many different ways, i.e., using semi-analytical methods [Chiron et al. 2019]. None of these methods can be directly applied to adaptive SPH as they depend on some sampling of the boundary that needs to be adapted to the particle resolution. Moreover, none of the existing methods can handle small boundary features at or below particle resolution properly, i.e., integral-based formulations smooth them out and particle-based methods yield uneven boundaries.

In our paper, we propose a novel-semi analytic boundary handling approach that comprises the following major contributions:

- An analytic, scale independent, solution to the interaction of particles with a planar boundary.
- A signed distance field based approach that handles the interaction of a particle with an arbitrary boundary by local planar approximations.
- Significantly improved handling of small boundary geometries at, or even below, the current particle resolution.
- A boundary interaction scheme based on a single contact that allows for easy pressure estimates and two-way coupling.

Our boundary handling scheme can be easily integrated into existing simulation methods and other boundary effects, e.g., friction, can be adapted without conceptual modifications. We evaluate our method in a variety of simulations to demonstrate its benefits compared to prior approaches, especially considering boundary representation.

2 RELATED WORK

In the following we will mostly focus on boundary handling methods in SPH and refer the reader to Koschier et al. [2019] for a wider overview of recent work in SPH in computer animation contexts.

Incompressibility plays an important role in realistic fluid simulations. Whereas initial work involved weakly compressible (WCSPH) methods [Müller et al. 2003] and the predictive-corrective incompressible (PCISPH) method [Solenthaler and Pajarola 2009], recent work focuses on implicit formulations (IISPH) [Ihmsen et al. 2013]. Following IISPH, Bender and Koschier [2015] proposed the current state of the art approach of enforcing incompressibility by adding an additional divergence-free solver. Many recent advances, considering incompressibility and stability, have been achieved by incorporating the boundary representation more directly into the SPH model, i.e., by pressure extrapolation [Band et al. 2018b], or by using particles for rigid-rigid interactions for strong rigid coupling (ILSPH) [Gissler et al. 2019]. Macklin and Müller [2013] developed a different SPH simulation method utilizing position-based dynamics (PBD), which was later integrated into a unified simulation model for fluid and rigid systems [Macklin et al. 2014].Moreover, spatially adaptive methods have been widely researched in the past, for example by Adams et al. [2007] for WCSPH, by Solenthaler and Gross [2011] using non mass-preserving approaches and recently by Winchenbach et al. [2017] for incompressible SPH methods. However, these methods usually assume that a (nearly) fixed particle resolution is in contact with the boundary.

Boundary handling in SPH has been a longstanding research topic and various different approaches have been proposed to handle fluid-boundary interactions in the last decade. In general, there exist four categories of boundary handling approaches:

(1) External boundary handling methods
(2) Particle-based methods
(3) Wall-renormalization methods
(4) Boundary-integral methods

External boundary handling methods were initially developed for weakly compressible SPH (WCSPH) utilizing a variety of approaches, i.e., particle level sets [Losasso et al. 2008] or direct forcing [Becker et al. 2009], however these methods are limited in their applicability as they cannot be tightly integrated into SPH methods.

Particle-based methods either handle boundary interactions using ghost particles representing boundary objects [Adami et al. 2012], or using explicitly sampled boundary particles placed on the surface of a boundary object[Akinci et al. 2012]. However, particle-based methods suffer from a strong dependence on the quality of the sampling. Band et al. [2017] proposed a particle-based boundary representation with improved accuracy in flat regions, by using local planar boundary fitting and resampling of the particle representation to reduce the sampling problems inherent to particle-based representations. Band et al. [2018a] proposed an extended boundary handling scheme that directly estimates pressure values on boundary particles, which has been further optimized by a moving-least-squares-based pressure extrapolation [Band et al. 2018b].

Wall-renormalization approaches extend the fluid domain into the boundary domain and were initially proposed by Feldman and Bonet [2007]. Ferrand et al. [2013] utilized this approach to realize a semi-analytical boundary handling scheme in 2D, which was later extended to 3D by Maryhofer et al. [2015]. Leroy et al. [2014] further extended this approach to handle incompressible SPH methods. However, these methods often only apply to 2D or viscous flows and can typically not be applied directly to adaptive simulations [Chiron et al. 2019]. Furthermore, these approaches cannot simply be integrated into computer animation approaches directly, as these approaches require an additional renormalization term on top of the direct summation of contributions.

A semi-analytical *boundary-integral method* was proposed by Fujisawa and Kenjiro [2015], that utilized empirically derived functions

to allow for incompressible SPH simulations in simple one-way coupled scenarios. Recently, Koschier and Bender [2017] developed a Boundary-integral method that precalculates boundary integrals on a fixed numerical grid, using an expensive pre-calculation method. This was improved by Bender et al. [2019] by pre-calculating a modified volume term. However, these methods either require expensive pre-computations or rely on empirically derived functions.

3 FOUNDATIONS OF BOUNDARY HANDLING IN SPH

After a short introduction to the fundamental aspects of SPH, this section gives a brief conceptual overview of the various boundary handling approaches, and how they affect SPH. This serves as a basis for the derivation of our method in the upcoming sections.

The underlying basis of SPH is an approximation of an integral identity using smoothing kernels with compact support radii. The integral identity of a function $A : \Omega \rightarrow \mathbb{R}^m$, defined over a domain $\Omega \subset \mathbb{R}^n$, can be written, using the Dirac delta function δ [Gingold and Monaghan 1977], as

$$A(\mathbf{x}) = \int_{\Omega} A(\mathbf{x}')\delta(\mathbf{x} - \mathbf{x}')d\mathbf{x}'. \tag{1}$$

Replacing the Dirac delta function with a radially symmetric smoothing kernel $W : \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$, defined over a spherical domain $\mathcal{D}_{\mathbf{x}}$, around $\mathbf{x}$ with radius h, commonly referred to as support radius, yields an approximation for A as

$$A(\mathbf{x}) \approx \int_{\mathcal{D}_{\mathbf{x}}} A(\mathbf{x}')W(|\mathbf{x} - \mathbf{x}'|, h)d\mathbf{x}'. \tag{2}$$

It is important to note that this approximation will only yield exact results, i.e., exactly representing a boundary geometry, if the support radius approaches 0, and thus (2) yields (1) again. For all nonzero support radii, this approximation introduces a smoothing of any quantity over a spatial domain, limiting the achievable resolution of an SPH simulation for a given support radius. The smoothing kernel can be defined as [Dehnen and Aly 2012]

$$W(r, h) = \frac{1}{h^n} C_n \hat{W}\left(\frac{r}{h}\right), \tag{3}$$

where n denotes spatial dimensionality, C_n is a normalization factor such that $\int_{\mathcal{D}_{\mathbf{x}}} W(|\mathbf{x}' - \mathbf{x}|, h)d\mathbf{x}' = 1$ and $\hat{W}$ being the actual kernel. Various kernel functions exist, however, we will focus on the cubic spline kernel, as it is widely used in computer animation. The cubic spline kernel function has a normalization factor of $\frac{80}{7\pi}$ in 2D and $\frac{16}{\pi}$ in 3D with the kernel defined as [Dehnen and Aly 2012]

$$\hat{W}_{\text{spline}}(q) = [1 - q]_+^3 - 4\,[0.5 - q]_+^3, \tag{4}$$

with $[\cdot]_+ = \max(0, \cdot)$. We define $\mathcal{D}_{\mathbf{x}}$ as the local boundary domain as the overlap of the overall boundary domain Ω_B with the support domain, $\mathcal{D}_{\mathbf{x}}^B = \mathcal{D}_{\mathbf{x}} \cap \Omega_B$, as well as the local fluid domain, $\mathcal{D}_{\mathbf{x}}^F = \mathcal{D}_{\mathbf{x}} \setminus \mathcal{D}_{\mathbf{x}}^B$. If there is no local boundary domain, i.e., $\mathcal{D}_{\mathbf{x}}^B = \emptyset$, (2) can directly be discretized using fluid particles only, where each particle carries a mass m and has a density ρ, yielding [Monaghan 1992]

$$A(\mathbf{x}) \approx \sum_j A_j \frac{m_j}{\rho_j} W(|\mathbf{x}_j - \mathbf{x}|, h), \tag{5}$$

where j is the set of neighboring fluid particles within h, with respect to $\mathbf{x}$. For further details on this derivation, as well as gradient terms, we refer the reader to [Price 2012] and [Koschier et al. 2019].

However, this discretization is only applicable in the absence of any overlap with a boundary domain. The different categories of boundary handling methods (see Sec. 2) treat regions containing some boundary segment significantly differently.

External boundary handling methods do not discretize, or evaluate, (2) for boundary regions, i.e., they use (5) everywhere and enforce boundary handling through some external mechanism, e.g., by clamping particle positions. These methods are not commonly used in modern simulation methods as they cannot be tightly integrated into an SPH method like IISPH or DFSPH.

Particle-based methods involve sampling the surface of a boundary domain with particles [Akinci et al. 2012]. Here each boundary particle b has a rest volume $V_b^0 = \frac{\gamma}{\sum_{b_b} W_{bb_b}}$ [Band et al. 2018a], depending on neighboring boundary particles b_b. This γ factor is used to correct for the lack of particles sampling the interior of the boundary and is commonly set to 0.7 to prevent penetration [Gissler et al. 2019]. This yields a modified SPH estimate including boundary particles as

$$A(\mathbf{x}) \approx \sum_j A_j \frac{m_j}{\rho_j} W_{ij} + \sum_b A_b \frac{V_b^0}{\delta_b} W_{ib}, \tag{6}$$

with $\delta_b = \sum_{b_f} V_{b_f} W_{bb_f} + \sum_{b_b} V_{b_b}^0 W_{bb_b} + \beta$. This formulation allows a natural extension of many SPH processes to boundary interactions and also enables strong rigid-fluid coupling [Gissler et al. 2019]. However, these interactions strongly depend on the resolution and quality of the sampling and are not applicable to spatially adaptive methods. Furthermore, sampling all regions of a boundary domain, even if they only briefly come into contact with fluid particles, can require significant amounts of extraneous boundary particles.

Wall-renormalization methods, commonly found in CFD contexts, determine a wall-renormalization term γ [Leroy et al. 2014] that describes the relation between $\mathcal{D}_{\mathbf{x}}$ and $\mathcal{D}_{\mathbf{x}}^F$, as

$$\gamma(\mathbf{x}) = \int_{\mathcal{D}_{\mathbf{x}}^F} W(|\mathbf{x} - \mathbf{x}'|, h)d\mathbf{x}', \tag{7}$$

This term is commonly evaluated using the divergence theorem, i.e., by evaluating (7) over the surface of the local boundary domain $\partial\mathcal{D}_{\mathbf{x}}^B$ using semi-analytical approaches; see [Chiron et al. 2019]. This yields a modified SPH estimate as

$$A(\mathbf{x}) \approx \frac{1}{\gamma(\mathbf{x})} \sum_j A_j \frac{m_j}{\rho_j} W_{ij}. \tag{8}$$

The evaluation of wall-renormalization methods, i.e., due to the γ term, is computationally expensive and assumes that the fluid domain can be extended into the boundary domain. The latter is, however, not always correct, for instance in case of a moving boundary object with a velocity independent of the fluid.

Boundary-integral methods directly evaluate the integral (2) over the boundary domain [Koschier and Bender 2017], i.e.,

$$A(\mathbf{x}) \approx \sum_j A_j \frac{m_j}{\rho_j} W_{ij} + A_b \int_{\mathcal{D}_{\mathbf{x}}^B} W(|\mathbf{x}_i - \mathbf{x}'|, h)d\mathbf{x}', \tag{9}$$

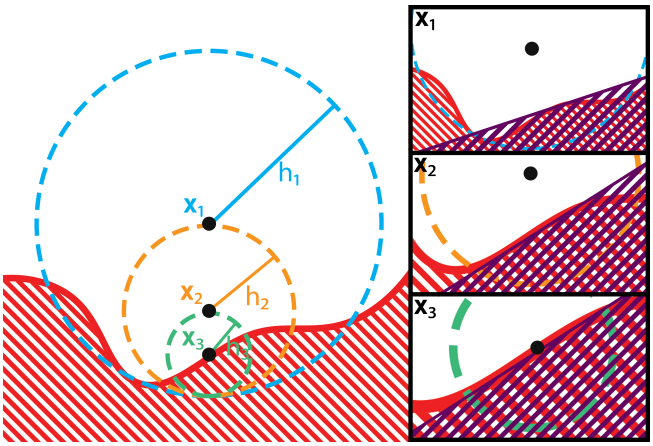


Fig. 2. Considering the boundary as locally flat is less accurate, i.e., for position $\mathbf{x}_1$ and support radius h_1(top right). However, with smaller support radii the assumption leads to much better representations (bottom right).

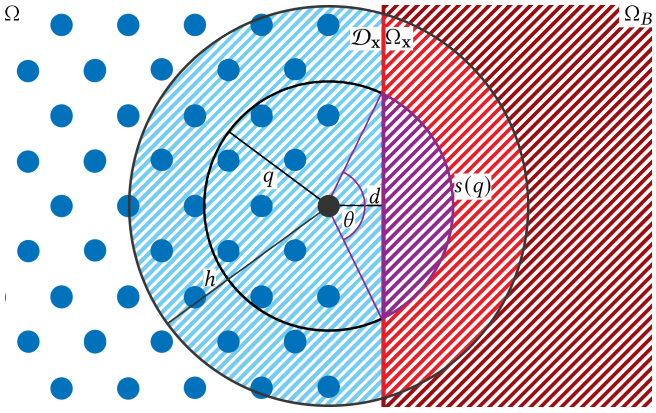


Fig. 3. This figure shows a planar boundary in 2D. Here a particle contains a fluid region, $\mathcal{D}_\mathbf{x}^F$, and a boundary region, $\mathcal{D}_\mathbf{x}^B$, within $\mathcal{D}_\mathbf{x}$. For a given relative distance q and a boundary distance d, we find an angle θ, which describes an arc segment $s(q)$, which we use to determine the integral.

with A_b representing the relevant quantity for the boundary domain. The boundary contribution $\lambda(\mathbf{x}) = \int_{\mathcal{D}_\mathbf{x}^B} W(|\mathbf{x}-\mathbf{x}'|, h)d\mathbf{x}'$ is either pre-calculated [Koschier and Bender 2017] or approximated [Fujisawa and Miura 2015]. Note that, as (2) is equal to 1 over $\mathcal{D}_\mathbf{x}$, the wall-renormalization factor $\gamma(\mathbf{x})$ is $1-\lambda(\mathbf{x})$. Precomputed values for $\lambda(\mathbf{x})$ are only valid for a specific particle resolution, as the value depends on the support radius, and empirically-based approaches are not consistent across varying particle resolutions.

4 METHOD OVERVIEW

Our novel semi-analytic boundary-integral-based method is built on the idea of treating all boundaries as locally planar and utilizing an analytic solution for $\lambda(\mathbf{x})$ to evaluate the contributions of all boundaries. This requires us to find a locally planar boundary, representing an overall boundary domain, for each boundary domain that a single particle interacts with. This means that (9) becomes

$$A(\mathbf{x}) \approx \sum_j A_j \frac{m_j}{\rho_j} W_{ij} + \sum_b A_b \lambda_{i,n}^b, \tag{10}$$

where b are the boundary domains intersecting $\mathcal{D}_\mathbf{x}$ and $\lambda_{i,n}^b$ describes the interaction of particle i with the planar boundary representing b in n dimensions for the position $\mathbf{x}_i$.

The assumption of local flatness, as shown in Fig. 2, works well for smooth boundaries where the boundary surface has a well defined first derivative at all positions, which can readily be utilized to find a local plane that becomes a better approximation as the support radius shrinks. Spatially adaptive methods can be used to ensure that high resolution particles are placed in areas of high boundary curvature to improve this approximation further.

Conceptually, directly evaluating (2) leads to a smoothing effect for sharp features, i.e., features that are as large as, or smaller than, the support radius are not properly represented. This smoothing causes the boundary shape to not be properly represented, e.g., sharp corners become smoothed out, which results in a strong dependence of boundary representation on particle resolution. Our method avoids this by ensuring that the locally planar representation of a boundary coincides with the actual boundary geometry such that $\int_{\mathcal{D}_\mathbf{x}^B} W(|\mathbf{x}-\mathbf{x}'|, h)d\mathbf{x}' = 0.5$, on the boundary surface. This means that particles, regardless of the geometry of the boundary domain, or the particle resolution, will be influenced solely based on the distance to the boundary surface, which in turn means that boundary features are not smoothed out. We will evaluate the differences in boundary-representation and how they influence the simulation, especially with regard to small features, in Sec. 9.2.

5 ANALYTIC PLANAR BOUNDARIES

To find an analytic solution for planar boundaries, we initially assume that particles have a support radius of 1 and demonstrate the analytic solutions in 2D and 3D; see Sec. 5.1 and 5.2 respectively. Sec. 5.3 extends the solution to arbitrary support radii. Sec. 5.4 discusses gradients and related terms. Sec. 5.5 discusses penalty terms.

5.1 2D Analytical Boundary Integrals

The boundary contribution term in 2D, for a planar boundary, only depends on the distance to the plane, which means that

$$\lambda_2(d) = \int_0^1 \int_0^{s(q)} C_2 \hat{W}(q) dl dq, \tag{11}$$

for spherical coordinates q and l, where $s(q)$ is the length of an arc with radius q, centered d away from the boundary, that is within $\mathcal{D}_\mathbf{x}^B$, i.e., $s(q) = 2q\pi$ for $\mathcal{D}_\mathbf{x}^B = \mathcal{D}_\mathbf{x}$. As we assume $\mathcal{D}_\mathbf{x}^B$ to be locally flat, $s(q)$ is the arc length of a circle segment C, based on the signed distance d from $\mathbf{x}$ to $\mathcal{D}_\mathbf{x}^B$; see Fig. 3. Using standard algebra we get

$$s(q) = 2q \operatorname{acos}\left(\frac{d}{q}\right), \tag{12}$$

for $d > 0$. If $q < d$ the arc length $s(q)$ would result in a complex number and as such we exclude this region by increasing the lower

bound for q to be at least d, for now. Inserting this into (11) yields

$$\int_d^1 \int_0^{2q\,\mathrm{acos}\left(\frac{d}{q}\right)} C_2\hat{W}(q)dldq = \int_d^1 C_2\hat{W}(q)2q\,\mathrm{acos}\left(\frac{d}{q}\right)dq. \tag{13}$$

We can readily extend this integral for $d < 0$ using the rotational symmetry of the smoothing kernel as

$$\lambda_2(d) = \int_0^1 C_2\hat{W}(q)2\pi q^2 dq - \lambda_2(-d) = 1 - \lambda_2(-d). \tag{14}$$

In order to evaluate the integral in (13) we have to specify the kernel being used. Therefore, we employ the cubic spline kernel function (4), although any other integrable smoothing kernel can be used. As the cubic spline kernel is defined piecewise, we need to split the integral for $d \leq 0.5$ into two definite integrals over $[d, 0.5]$ and $(0.5, 1]$, whereas for $d > 0.5$ we only need to consider the definite integral over $[d, 1.0]$. These definite integrals, shown in the appendix as (56), can be solved symbolically, and the final expression for $\lambda_2(d)$ is given in the Appendix in (57).

5.2 3D Analytical Boundary Integrals

In 3D, we analogously rewrite the boundary contribution term as an integral over the 3D unit ball V, using spherical coordinates, as

$$\lambda_3(d) = \iiint_V C_3\hat{W}(q)dV = \int_0^1 \iint_0^{A(q)} C_3\hat{W}(q)dAdq, \tag{15}$$

where $A(q)$ denotes the partial area of a sphere of radius q inside the boundary domain $\mathcal{D}_{\mathbf{x}}^B$, i.e., $A(q) = 4\pi q^2$ for $\mathcal{D}_{\mathbf{x}}^B = \mathcal{D}_{\mathbf{x}}$. The relevant area is given as the curved section of a spherical cap C at distance d from the center for a sphere of radius q, using standard algebra, as

$$A(q, d) = 2\pi q(q - d), \tag{16}$$

which, again, is only well defined for $q \geq d$ and we, therefore, limit the integral to $[d, 1]$. Inserting (16) into (15) then yields

$$\int_d^1 \iint_0^{2\pi q(q-d)} C_3\hat{W}(q)dAdq = \int_d^1 C_3\hat{W}(q)2\pi q(q-d)dq. \tag{17}$$

Analogously to 2D, we split the integral-based on the piecewise definition of the cubic spline kernel, and add an extension for $d < 0$ using rotational symmetry, yielding the definite integrals shown in the Appendix as (59). These can then be evaluated symbolically, yielding (60) in the Appendix, which put back into $\lambda_3(d)$ result in the boundary contribution term in 3D for a planar boundary as

$$\lambda_3(d) = \begin{cases} \frac{1}{60}\left[192d^6 - 288d^5 + 160d^3 - 84d + 30\right], & d \in [0, 0.5] \\ \frac{-8}{15}\left[2d^6 - 9d^5 + 15d^4 - 10d^3 + 3d - 1\right], & d \in (0.5, 1], \\ 1 - \lambda_3(-d), & d \in [-1, 0). \end{cases} \tag{18}$$

Interestingly, this term is significantly more simple than the equivalent term in 2D, see (57), but yields very similar numerical values.

5.3 Scaling factor

In order to extend the prior derivations to non-unit support radii we insert the definition of a generic kernel function from (3), into an integral over an arbitrary boundary domain $\mathcal{D}_{\mathbf{x}}^B$, which yields

$$\int_{\mathcal{D}_{\mathbf{x}}^B} W(r, h)d\mathbf{x}' = \int_{\mathcal{D}_{\mathbf{x}}^B} \frac{1}{h^n} C_n\hat{W}\left(\frac{r}{h}\right)d\mathbf{x}'. \tag{19}$$

As $\mathcal{D}_{\mathbf{x}}^B$ includes all radii from 0 to h, we perform a u-substitution with $q = r/h$, which yields

$$\int_{\mathcal{D}_{\mathbf{x}}^B} \frac{1}{h^n} C_n\hat{W}\left(\frac{r}{h}\right)d\mathbf{x}' = \int_{\mathcal{D}_{\mathbf{x}}^{B*}} C_d\hat{W}(q)d\mathbf{x}^*, \tag{20}$$

where the factor $\frac{1}{h^n}$ vanishes due to the substitution and $\mathcal{D}_{\mathbf{x}}^{B*}$ is the result of scaling $\mathcal{D}_{\mathbf{x}}^B$ to a radius of 1, which also means that the flat boundary is at a distance $d^* = d/h$ instead of d and $\mathbf{x}^* = \frac{\mathbf{x}'}{h}$.

In addition to the integral of the kernel function over the boundary domain, integral values of other functions may have to be calculated, e.g., the source term of a pressure solver like DFSPH [Bender and Koschier 2015]. Commonly, these functions depend on a radially symmetric function $f : \mathbb{R}_0^+ \to \mathbb{R}^m$, based on the relative distance r/h, divided by some power of the support radius, which yields an integral of the form

$$A_b \int_{\mathcal{D}_{\mathbf{x}}^B} \frac{1}{h^s} f\left(\frac{r}{h}\right)d\mathbf{x}'. \tag{21}$$

Here we can, again, apply a u-substitution to yield

$$A_b \int_{\mathcal{D}_{\mathbf{x}}^{B*}} \frac{1}{h^{s-n}} f(q)d\mathbf{x}^*. \tag{22}$$

5.4 Gradient Terms

In addition to the boundary contribution term λ_n, related terms, i.e., the spatial derivative, ∇_i, with respect to a particle i, are required, for example for pressure forces. In order to calculate these gradients, we use the fact that the spatial derivative of the distance d for a flat boundary $\mathcal{D}_{\mathbf{x}}^B$ equals the normal of the flat boundary $\mathbf{n}_b$, which when normalized yields $\hat{n}_b = \frac{\nabla_i d}{||\nabla_i d||}$. Applying the chain rule gets

$$\nabla_i\left(A_b\lambda_n^b\right) = A_b\nabla_i\lambda_n^b(q) = A_b\frac{1}{h}\hat{n}_b\frac{\partial\lambda_n^b(q)}{\partial q}. \tag{23}$$

However, these direct gradient terms in SPH are not exact for constant functions and do not preserve symmetric interactions. By following the ideas of Price [2012] for particles, we can yield analogous terms for our integral-based boundary representation; see (66) and (67) in the Appendix. Applying $\frac{\partial}{\partial q}$ to $\lambda_3^b(q)$ yields

$$\frac{\partial\lambda_3(q)}{\partial q} = \begin{cases} \frac{3}{15}\left[96q^5 - 120q^4 + 40q^2 - 7\right], & q \in [0, 0.5] \\ \frac{-24}{15}\left[4q^5 - 15q^4 + 20q^3 - 10q^2 + 1\right], & q \in (0.5, 1], \\ -\frac{\partial}{\partial q}\lambda_3(-q), & q \in [-1, 0). \end{cases} \tag{24}$$

In order to realize two-way coupling, modern pressure solvers, see Bender and Koschier [2015] Eqn. (8), evaluate a factor that, in terms of boundary integrals, is expressed as

$$\frac{\alpha_i}{\rho_0^2} = \left\|\int_{\Omega_{\mathbf{x}}} \nabla_i W(|\mathbf{x_i} - \mathbf{x}|, h)d\mathbf{x}\right\|^2 + \int_{\Omega_{\mathbf{x}}} ||\nabla_i W(|\mathbf{x_i} - \mathbf{x}|, h)||^2 d\mathbf{x}, \tag{25}$$

where the former term can be directly calculated using the gradient of the boundary contribution; see (23). To calculate the latter term, referred to as $\mathcal{S}$, we apply the spatial derivative ∇_i to (3), yielding

$$\nabla_i W(|\mathbf{x}_j - \mathbf{x}_i|, h) = \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|}\frac{1}{h^{n+1}} C_n\frac{\partial\hat{W}(q)}{\partial q}. \tag{26}$$

Consequently, we get

$$\mathcal{S}_n(d) = \int_{\mathcal{D}_\mathbf{x}^B} \frac{1}{h^{2n+2}} C_n^2 \left[\frac{\partial \hat{W}(q)}{\partial q}\right]^2 d\mathbf{x}', \tag{27}$$

Note that in contrast to λ_n in Eqn. (56) and (59), the integration of the function over the whole domain $\mathcal{D}_\mathbf{x}$ is not equal to 1. This results in the definite integrals (61), which can be symbolically integrated; see (62) in the Appendix. To correct for scaling, the result needs to be divided by h^{n+2} (Sec. 5.3), which results in a final term for $\mathcal{S}_3(d)$; see (63) in the Appendix. Similar steps could be taken to determine the Laplacian over a boundary, but these terms find no practical use in modern pressure solvers and as such are left out for brevity.

5.5 Penalty Terms

A problem with the analytical solution proposed thus far is that for a particle that is fully within the boundary, i.e., $\mathcal{D}_\mathbf{x} = \mathcal{D}_\mathbf{x}^B$, the density gradient will be zero. This means that if a particle fully penetrates the rigid body it will never get repelled out of the rigid body. To avoid this problem Koschier and Bender [2017] and Bender et al. [2019] proposed different penalty functions, which are applied to the boundary contribution to penalize full penetration by artificially adding a monotonically increasing term onto the boundary terms. Similar to Koschier and Bender [2017], we chose the following penalty function

$$\beta(d) = 1 - \frac{d}{h}, \frac{\partial \beta(d)}{\partial d} = -h^{-1}. \tag{28}$$

We directly multiply this penalty term onto $\lambda_n(d)$, which yields

$$\lambda_n^\beta(d) = \beta(d)\lambda_n(d), \tag{29}$$

which for gradients, by applying the product rule, yields

$$\nabla_i \lambda_n^\beta(d) = \frac{\partial \beta}{\partial d} \hat{n}_{\Omega_\mathbf{x}} \lambda_n(d) + \beta(d) \nabla_i \lambda_n(d). \tag{30}$$

6 ARBITRARY BOUNDARY HANDLING

To find a locally planar representation of a boundary, to which we can apply the analytic solution demonstrated in Sec. 5, we propose to utilize a signed distance field (SDF). Signed distance fields can be used to represent arbitrary geometries and yield the same signed distance value and gradient, regardless of particle resolution, making them well suited for adaptive methods. However, SDFs, in general, are not ideally suited for highly concave objects, as particles would only be influenced by one part of the geometry, i.e., they are limited to single contact points. This limitation, however, can be resolved by performing a convex body decomposition on the boundary geometry that allows multiple points of contact for a single particle with the same overall boundary geometry. Furthermore, *discretized* SDFs might not accurately represent the boundary geometry, e.g., due to aliasing artifacts. While Sec 9.5 discusses how this induced error can be measured, and reduced. We assume a continuous SDF for this derivation, i.e., an SDF that is not limited in accuracy.

An SDF Φ_B is determined based on the surface of a boundary domain ∂B, for at an arbitrary spatial position $\mathbf{x}$, as

$$\Phi_B(\mathbf{x}) = s(\mathbf{x}) \inf_{\mathbf{x}^* \in \partial B} ||\mathbf{x} - \mathbf{x}^*||, s(\mathbf{x}) = \begin{cases} -1, & \mathbf{x} \in B \\ 1, & \text{else.} \end{cases} \tag{31}$$

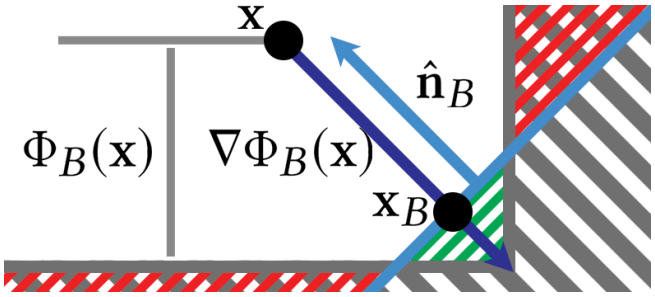
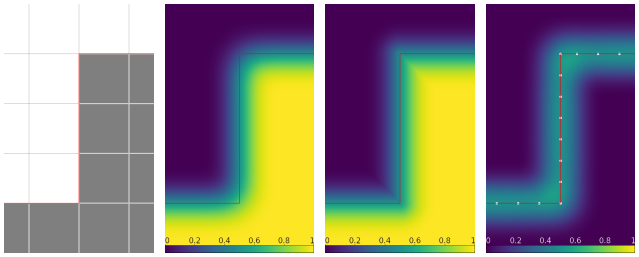


Fig. 4. The construction of a local plane (blue) for a concave corner of a boundary (gray) based on the signed distance field Φ_B. This constructed plane ignores the contribution of some regions (red), but additionally includes other regions (green), causing an overall underestimate.



(a) Boundary (b) Numerical (c) Distance-based (d) Particle-based

Fig. 5. The boundary contribution term λ evaluated (a) using a numerical solution (b), our signed distance field-based solution (c) and a particle-based representation (d). The signed distance field-based solution represents the boundary geometry well, but shows underestimates in concave regions and overestimates in convex regions. The particle-based approach cannot distinguish between inside and outside of the boundary and shows similar overestimates in convex regions and underestimates in concave regions. Note that the particle-based approach only represents the surface of the boundary, i.e., there is no way to distinguish interior and exterior of the boundary. λ color coded from 0 (purple) to 1 (yellow).

Using the normalized numerical gradient of the signed distance field, e.g., determined using a central difference scheme, we can find a unit normal $\hat{\mathbf{n}}_B = \nabla\Phi_B(\mathbf{x})/|\nabla\Phi_B(\mathbf{x})|$, which yields an approximate closest point on the boundary surface as

$$\mathbf{x}_B = \mathbf{x} - \Phi_B(\mathbf{x})\hat{\mathbf{n}}_B. \tag{32}$$

Note that the usage of a numerical gradient here results in a different result than using a continuous gradient. Fig. 4 shows how for a point $\mathbf{x}$ sitting on the angle dividing between two planes we find a numerical gradient pointing away from the corner, which also results in a contact point $\mathbf{x}_B$ that does not lie on the actual boundary geometry, but instead gives an approximative representation of the corner. Using $\hat{\mathbf{n}}_B$ and $\mathbf{x}_B$ we then construct a plane E_b, which represents the boundary b, in normal form

$$E_b : \hat{\mathbf{n}}_B \cdot (\mathbf{x} - \mathbf{x}_B) = 0, \tag{33}$$

where basing $\hat{\mathbf{n}}_B$ on the position $\mathbf{x}$ instead of $\mathbf{x}_B$ ensures that forces act away from the boundary. Figure 4 depicts this configuration

for a convex corner, where the constructed plane causes an underestimate of the boundary contribution term. This underestimate, however, is unavoidable when trying to replace a complex boundary by a single plane. The effect of this underestimate is evaluated in the results section. For a concave configuration the approximative boundary plane causes an overestimate of the boundary contribution term. Fig. 5 shows the boundary density contribution for a simple boundary (Fig. 5a), using the 30th order quadrature rule of Xiao and Gambutas [2010] for a triangulated boundary domain (Fig. 5b), our signed distance field-based approach (Fig. 5c), and a particle-based approach (Fig. 5d). The particle-based approach also shows over and underestimates for the contribution and provides distinction between the outside and the inside of the boundary, causing severe problems for fluid particles penetrating the boundary slightly.

For moving boundary objects, a recalculation of the signed distance field in every simulation step would be far too expensive. Therefore, we use a rigid body transformation matrix $\mathbf{M}_B$ from local model coordinates, in which the origin is the object's center of mass, to global simulation coordinates. Using $\mathbf{M}_B$ any point $\mathbf{x}$, in global fluid space, can be transformed, into the local model space, via

$$\mathbf{x}^* = \mathbf{M}_B^{-1}\mathbf{x}. \tag{34}$$

Afterwards, the closest point on the boundary $\mathbf{x}_B^*$ is constructed in model space and transformed back to global simulation coordinates using $\mathbf{M}_B$. The corresponding normal of the plane $\hat{\mathbf{n}}_B^*$ is calculated using the transposed inverse transformation $\mathbf{M}^{-T}$, yielding

$$E_b : \left(\mathbf{M}_B^{-T}\hat{\mathbf{n}}_B^*\right) \cdot \left(\mathbf{x} - \mathbf{M}_B\mathbf{x}_B^*\right) = 0, \tag{35}$$

as the representative plane with the contact point $\mathbf{M}_B\mathbf{x}_B^*$. This process is also depicted in Algorithm 1. We also utilize this process in case of multiple boundary objects, i.e., each boundary object is represented by an indepedent SDF in the respective objects model coordinates, and interacting with multiple boundary objects only requires a transformation into each boundary objects coordinate system. Accordingly, our approach can readily handle multiple boundary objects without requiring complex SDF operations, i.e., finding a global SDF describing all objects in a scene.

ALGORITHM 1: The process to find a locally representative plane for a complex boundary object B using its signed distance field in model coordinates.

$\mathbf{x}^* \leftarrow \mathbf{M}_{\mathcal{B}}^{-1}\mathbf{x}$
$d \leftarrow \Phi_B^*(\mathbf{x}^*)$
If $d > h$
  Return
$\hat{\mathbf{n}}_B^* \leftarrow= -\nabla\Phi^*(\mathbf{x}^*)/|\nabla\Phi^*(\mathbf{x}^*)|$
$\mathbf{x}_B^* \leftarrow \mathbf{x}^* - \Phi^*(\mathbf{x}^*)\hat{\mathbf{n}}_B^*$
Return the representative plane (35)

In order to implement two-way coupling into our simulation, we require several properties of the rigid body b based on the contact point $\mathbf{x}_{ib}$ associated with the position of a fluid particle i. Therefore, we first determine the center of mass of the rigid body in simulation space as $\mathbf{x}_b^c = \mathbf{M}_b\mathbf{0}$, which yields the velocity of the contact point as

$$\mathbf{v_{ib}} = \mathbf{v}_b + \omega_b \times \left(\mathbf{x}_i - \mathbf{x}_b^c\right), \tag{36}$$

where $\mathbf{v}_b$ and ω_b are the linear and angular velocity of the overall boundary object. The acceleration $\mathbf{a_{ib}}$ of the contact point can similarly be calculated. Determining the influence of a boundary object on a particle can be done straightforwardly by including the boundary contribution term for a particle (see Eq. 10), however determining the influence of particles on the boundary cannot be done directly. Instead, we propose to add up the inverse force from a boundary, $\mathbf{F}_{i\to b} = -\mathbf{F}_{b\to i}$, due to all particles, which yields the influence of the fluid on the boundary. In addition to the force $\mathbf{F}_{i\to b}$ we also require the torque as

$$\tau_{i\to B} = m_i\mathbf{a}_{i\to B} \times \left[\mathbf{x} - \mathbf{x}_c\right]. \tag{37}$$

From this, we determine the linear and angular acceleration of the overall boundary object, based on the inertia matrix $\mathbf{I}_b^*$ in model coordinates as

$$a_{i\to B} = \mathbf{F}_{i\to B}/m_B, \alpha_{i\to B} = \mathbf{M}_B\left(\mathbf{I}^*\right)_B^{-1}\mathbf{M}_B^{-1}\tau_{i\to B}. \tag{38}$$

Finally, for implementing our method, the maximum permissible timestep needs to be considered. We utilize the CFL condition [Ihmsen et al. 2014], excluding shock terms, as

$$\Delta t^{\max} = 0.4\max_i \frac{||\mathbf{v}_i||}{h_i}. \tag{39}$$

For our semi-analytical approach this is not directly possible as there are no boundary particles that we can use (39) for. Instead, we utilize the point furthest from the axis of rotation of a boundary object b, $\mathbf{x}_{\mathrm{b,max}}$, in order to determine the maximum local velocity of the boundary object as

$$\mathbf{v}_{\mathrm{b,max}} = \mathbf{v}_b + \omega_b \times \left[\mathbf{x}_{\mathrm{b,max}} - \mathbf{x}_b^c\right], \tag{40}$$

based on which we can determine the maximum permissible timestep analogous to Eq. (39). However, instead of the support radius for the particle, we use the smallest support radius of any particle in the simulation to ensure stability in all cases.

7 INTEGRATION INTO DFSPH

In this section we discuss the integration of our boundary integral method with a divergence-free SPH method based on Bender and Koschier [2015] and Band et al. [2018b]; see Algorithm 2. The first step of the simulation comprises the calculation of the density value for each fluid particle i including boundary contributions, utilizing (10), as

$$\rho_i = \sum_j m_j W_{ij} + \sum_b \rho_b \lambda_{i,n}^b. \tag{41}$$

For boundary objects we assume that ρ_b is equal to the object's respective rest density, as we only allow rigid body transformations, which only change the position of an object but not its shape. Note that V_i here refers to the apparent volume of a fluid particle

$$V_i = \frac{m_i}{\rho_i}, \tag{42}$$

instead of the actual volume $V_i^0 = \frac{m_i}{\rho_0}$; see Band et al. [2018a]. We also require a predicted velocity, $\mathbf{v}_i^* = \mathbf{v}_i + \frac{\Delta t}{m_i}\mathbf{F}_i^{\mathrm{adv}}$, after taking advection forces into account. In order to calculate this predicted velocity for the boundary objects, we utilize standard rigid body physics to find the specific local boundary velocity $\mathbf{v}_{ib}^*$ for a particle,

which depends on the particle's contact point $\mathbf{x}_{ib}$ at the boundary. Next we need to determine the diagonal element α_i [Band et al. 2018b] as

$$\begin{aligned}\alpha_i = &- V_i \Delta t^2 \frac{1}{m_i} || \sum_j V_j \nabla_i W_{ij} + \sum_b \nabla \lambda^b_{i,n} ||^2 \\ &- V_i \Delta t^2 \sum_j \frac{V_j^2}{m_j} ||\nabla_i W_{ij}||^2 - V_i \Delta t^2 \sum_b \frac{1}{\rho_b} \mathcal{S}^b_{i,n},\end{aligned} \tag{43}$$

where $\mathcal{S}^b_n$ is the given in Eq. (63) for $n = 3$. For all non two-way coupled and static boundary objects $\mathcal{S}^b_n$ is set to 0, as no pressure forces are applied to these boundaries; see also [Band et al. 2018a]. Similar to Band et al. [2018b], we determine the source terms for the divergence-free and incompressible solver as

$$s_i^{\text{inc}} = 1 - \frac{m_i}{V_i} - \Delta t \nabla_i \cdot \mathbf{v}^*, \quad s_i^{\text{div}} = -\Delta t \nabla_i \cdot \mathbf{v}, \tag{44}$$

respectively. The velocity divergence can be calculated as

$$\nabla_i \cdot \mathbf{v} = \sum_j V_j \Delta \mathbf{v}_{ij} \cdot \nabla_i W_{ij} + \sum_b \Delta \mathbf{v}_{ib} \cdot \nabla \lambda^b_{i,n}, \tag{45}$$

where $\Delta \mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$. Note that we only enforce the divergence-freedom for fluid-fluid interactions to avoid excessive adhesion and repulsion as a particle separating from a boundary would cause negative divergence, i.e., its density becomes lower, and a particle approaching a boundary would cause positive divergence. This is because we cannot evaluate the opposite effect on the boundary, i.e., when a particle has positive divergence the boundary object would experience negative divergence, during the divergence-free solving process. Other boundary-integral methods, e.g., [Koschier and Bender 2017], would be affected by the same problem. To initialize the pressure, we follow the proposed initialization of Gissler et al. [2019] and set $p_i = 0$ for all fluid particles. The next step is to determine the acceleration of a fluid particle due to pressure forces, due to fluid-fluid interactions, as [Bender and Koschier 2015]

$$a^p_{\text{fluid}\to i} = -\sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla_i W_{ij}. \tag{46}$$

Using the same formulation to calculate the pressure acceleration imposed by rigid objects requires a pressure value p_{ib} for the boundary. This pressure value is calculated by determining the contact point $\mathbf{x}_{ib}$ on the boundary and performing the moving-least-squares-based pressure extrapolation method proposed by Band et al. [2018b]. As we assume quantities to be locally constant, we can directly use the pressure value p_{ib} for the whole interaction. Thus, the total acceleration imposed on a fluid particle i by all rigid objects can be determined as

$$\mathbf{a}^p_{\text{rigid}\to i} = -\sum_b \rho_b \left(\frac{p_i}{\rho_i^2} + \frac{p_{ib}}{\rho_b^2} \right) \nabla_i \lambda^b_{i,n}. \tag{47}$$

We additionally utilize the acceleration imposed on a fluid particle by a single rigid object b, $\mathbf{a}^p_{b\to i}$, to calculate the reflected force on b, $\mathbf{F}^p_{i\to b} = -m_i \mathbf{a}^p_{b\to i}$, to realize two-way coupling effects. Accumulating the coupled force imposed by all particles yields the acceleration of the rigid object as a whole, which yields the local boundary acceleration $\mathbf{a}^p_{ib}$, at the contact point with respect to i. Moreover, we calculate the divergence of the estimated acceleration as

$$\nabla_i \cdot a_i^p = \Delta t^2 \sum_j V_j \mathbf{a}^p_{ij} \cdot \nabla_i W_{ij} + \Delta t^2 \sum_b \mathbf{a}^p_{ib} \cdot \nabla \lambda^b_{i,n}, \tag{48}$$

and an updated pressure value for each fluid particle as

$$p_i^* = p_i + \frac{\omega}{\alpha_i} \left(s_i - \nabla_i \cdot a_i^p \right), \tag{49}$$

where ω is the relaxation coefficient for a relaxed Jacobi solver, chosen as 0.5. We clamp the pressure to positive values for the incompressible solver, to avoid excessive surface-tension like effects. This process is repeated until the mass weighted average of the residual, $r_i = \nabla_i \cdot a_i^p - s_i$, converges below a threshold η, i.e.,

$$\sum_i r_i m_i \leq \eta \sum_i m_i, \tag{50}$$

where η is usually chosen as 0.0001 for the incompressible solver and 0.001 for the divergence-free solver, similar to Bender and Koschier [2015]. After the solver converges, we calculate the pressure acceleration and apply it to both fluid particles and boundary objects. We finally integrate the system accordingly.

ALGORITHM 2: Our overall simulation algorithm using a divergence-free and incompressible simulation approach with two-way coupling for complex boundary objects.

$\rho_i \leftarrow \sum_j m_j W_{ij} + \sum_b \rho_b \lambda^b_n$; $V_i \leftarrow \frac{m_i}{\rho_i}$ (41, 42)
Calculate Diagonal element α_i (43)
$p_i \leftarrow 0$; $s_i^{\text{div}} \leftarrow -\Delta t \nabla_i \cdot \mathbf{v}$ (44)
while $\sum_i m_i \nabla_i \cdot \mathbf{a}_i^p - s_i^{\text{div}} > \eta^{\text{div}} \sum_i m_i$ (50)
  Calculate Boundary pressure p_{ib} using MLS
  Calculate Acceleration due to pressure (47,46)
  $p_i^* \leftarrow p_i + \frac{\omega}{\alpha_i} \left(s_i - \nabla_i \cdot a_i^p \right)$, (49)
Calculate Final acceleration due to pressure $\mathbf{a}_i^{\text{div}}$ (47,46)
Calculate Non-pressure acceleration $\mathbf{a}_i^{\text{adv}}$
$\mathbf{v}_i^* \leftarrow \mathbf{v}_i + \Delta t \left[\mathbf{a}_i^{\text{adv}} + \mathbf{a}_i^{\text{div}} \right]$
$p_i \leftarrow 0$; $s_i^{\text{inc}} \leftarrow 1 - \frac{m_i}{V_i} - \Delta t \nabla_i \cdot \mathbf{v}^*$ (44)
while $\sum_i m_i \nabla_i \cdot \mathbf{a}_i^p - s_i^{\text{inc}} > \eta^{\text{inc}} \sum_i m_i$ (50)
  Calculate Boundary pressure p_{ib} using MLS
  Calculate Acceleration due to pressure (47,46)
  Calculate Acceleration of boundaries; see Sec. 6
  $p_i^* \leftarrow p_i + \frac{\omega}{\alpha_i} \left(s_i - \nabla_i \cdot a_i^p \right)$, (49)
Calculate Final acceleration due to pressure $\mathbf{a}_i^{\text{p}}$ (47,46)
Calculate Acceleration due to friction $\mathbf{a}_i^c$ (55)
$\mathbf{v}_i \leftarrow \mathbf{v}_i + \Delta t \left[\mathbf{a}_i^p + \mathbf{a}_i^{\text{div}} + \mathbf{a}_i^c + \mathbf{a}_i^{\text{adv}} \right]$; $\mathbf{x}_i \leftarrow \mathbf{x}_i \Delta t \mathbf{v}_i$
Update Rigid body system; see Sec. 6

8 FRICTION

To model fluid-rigid friction effects Koschier and Bender [2017] proposed a Coulomb model based force, which we also adopt for our approach as it is based on the same underlying boundary-integral model, however our method can also be applied to other friction models, e.g., [Rogers et al. 2010]. The friction model of Koschier and

Bender [2017], however, does not consider boundary pressure values. The Coulomb model determines a friction force for the interaction of two bodies i and b as

$$\mathbf{F}^c_{b\to i} \leq \mu ||\mathbf{F}^n_{i\to b}|| \hat{\mathbf{t}}_{ib}, \tag{51}$$

where μ is the friction coefficient, $||F^n_{i\to b}||$ the force imposed on b by i in normal direction and $\hat{\mathbf{t}}_{ib}$ the direction of the relative tangential velocity of i and b. The tangential velocity can be determined as

$$\mathbf{t}_{ib} = \Delta\mathbf{v}_{ib} - (\Delta\mathbf{v}_{ib} \cdot \hat{\mathbf{n}}_b) \cdot \hat{\mathbf{n}}_b, \tag{52}$$

with the relative velocity $\Delta\mathbf{v}_{ib} = \mathbf{v}_i - \mathbf{v}_b$. For the normal force we simply consider the pressure force, including the estimated boundary pressure of the rigid body, as

$$||\mathbf{F}^p_{b\to i}|| = -m_i \rho_b \left(\frac{p_i}{\rho_i^2} + \frac{p_{ib}}{\rho_b^2} \right) ||\nabla \lambda^b_{i,n}||. \tag{53}$$

The acceleration caused by all rigid bodies b on i then becomes

$$\mathbf{a}'_{b\to i} = \frac{1}{m_i} \sum_b \mathbf{F}^c_{b\to i} \tag{54}$$

The velocity change due to this imposed acceleration, however, can become greater than the relative velocity, which would cause unphysical behavior. To exclude such cases we limit the imposed acceleration to at most $-\frac{1}{\Delta t}\mathbf{t}_{ib}$ if $\Delta t \mathbf{a}'_{b\to i} \cdot \hat{\mathbf{t}}_{ib} \geq \Delta\mathbf{v}_{ib} \cdot \hat{\mathbf{t}}_{ib}$, which yields the final friction acceleration term as

$$\mathbf{a}^c_{b\to i} = \frac{1}{m_i} \sum_b \begin{cases} \mathbf{a}'_{b\to i}, & \Delta t \mathbf{a}'_{b\to i} \cdot \hat{\mathbf{t}}_{ib} < \Delta\mathbf{v}_{ib} \cdot \hat{\mathbf{t}}_{ib} \\ -\frac{1}{\Delta t}\mathbf{t}_{ib}, & \text{else.} \end{cases} \tag{55}$$

9 EVALUATION

In this section, we evaluate our method by comparing it against other boundary representations for small-sized boundary features in 2D (Sec. 9.2- 9.4), one-way and two-way coupled simulations in 3D (Sec. 9.6), for different boundary samplings (Sec. 9.7), as well as regarding adaptivity, friction and stability (Sec. 9.8-9.10).

9.1 Simulation Setup

We implemented our method in the open source GPU-based SPH framework openMaelstrom [Winchenbach 2019], using the adaptive method of Winchenbach et al. [2017], XSPH based artificial viscosity [Monaghan 2002], fluid-air phase interactions based on Gissler et al. [2017], surface tension based on Akinci et al. [2013a], adaptive time-stepping based on the CFL condition [Ihmsen et al. 2013], the underlying data structure of Winchenbach and Kolb [2019] and the cubic spline kernel [Monaghan 2005]. We used an incompressibility threshold of 0.01% and a divergence threshold of 0.1%, similar to Bender and Koschier [2015]. Surface distance calculations, for adaptivity, are based on Horvath and Solenthaler [2013]. For comparisons with particle based methods we used ILSPH [Gissler et al. 2019] with the optimized particle sampling of Bell et al. [2005] in 3D and the approach of Akinci et al. [2013b] with a regular sampling in 2D. The numerical boundary-integral method in 2D is realized by triangulating the boundary and using the quadrature rule of Xiao and Gambutas [2010] (at 30th order) to evaluate (2) over each triangle. The resulting images, utilizing this framework, were rendered using a Monte-Carlo based path-tracing algorithm. In our results frame-time refers to the accumulated runtime to simulate 16.67ms of simulation-time. For the comparison with the volume maps approach [Bender et al. 2019], we utilized the open source SPH framework SPlisHSPlasH [Bender 2016], using the same methods for fluid effects, e.g., surface tension, running on a CPU instead of on a GPU, and resulting images rendered using Autodesk Maya. All results were generated using a single Nvidia GeForce RTX 2080Ti GPU with 11 GiB of VRAM, an AMD 3970X, and 64 GiB of RAM.

In 2D we compute SDFs directly, without sampling, based on a polygon representing the boundary domain. In 3D SDFs can also be evaluated on the fly, based on a triangular mesh, but this is usually computationally prohibitive. Instead, we precompute a discrete hierarchical SDF using OpenVDB [Museth et al. 2013], based on the boundary mesh, and then transfer the SDF onto a GPU using GVDB [Hoetzlein 2016]. The process of constructing this SDF, and finding an appropriate grid resolution, is discussed in Sec. 9.5. The spatial gradients of the SDFs are determined using a second order central difference scheme. If this results in a zero length gradient, we instead utilize a first order forward difference scheme.

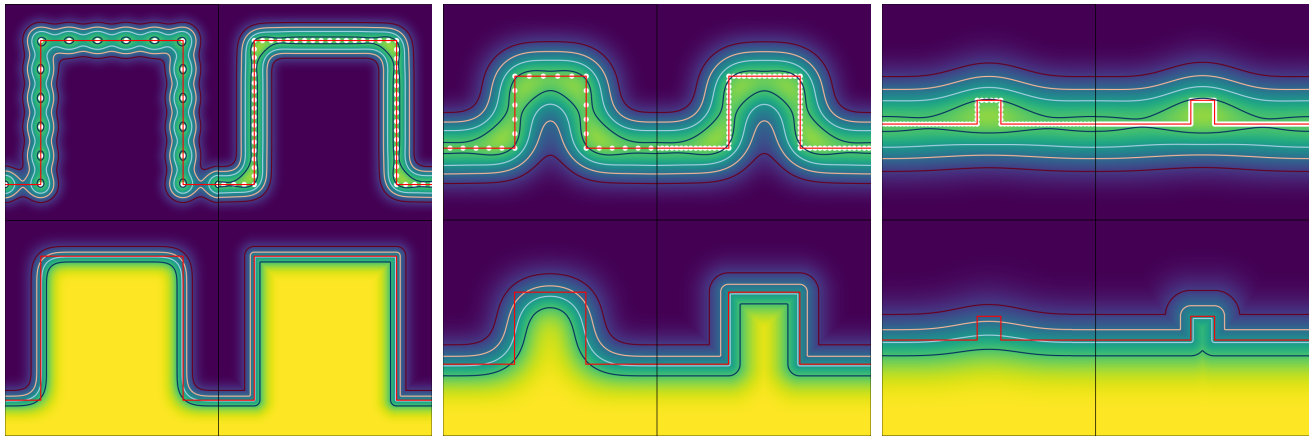
The matrix inverse for the moving-least-squares-based pressure estimation process [Band et al. 2018a] is evaluated using a singular value decomposition (SVD). In 2D we use a closed form solution for the SVD of a 2×2 matrix and the optimized SVD process for 3×3 matrices of McAdams et al. [2011] in 3D. The implementation of the 3D process is available as opensource code [Winchenbach 2018].

9.2 Boundary Representation

We compare our approach to a numerical boundary integral and a particle-based approach with a coarse and a fine boundary sampling, regarding their ability to represent a fixed size boundary feature with 1:4, 1:1 and 4:1 ratio of support radius to boundary feature size. We, therefore, compute isocontours where particles with no neighbors and quarter, half and three quarter full neighborhoods are located. For the fluid to interact correctly with a boundary feature, the isocontours of the fluid particles have to follow the geometry of the feature and must not intersect it. In this comparison the numerical boundary-integral approach yields identical results to a wall-renormalization approach.

For a 1:4 ratio (see Fig. 6 left), our approach and the boundary integral approach yield identical results in planar-regions h away from any corner. Close to corners, however, our approach generates significantly sharper isocontours, whereas the boundary-integral yields smoothed out corners. The particle-based approach, at a coarse sampling, cannot represent this ratio as the boundary would be penetrable for isolated particles, whereas a fine sampling of the boundary is impenetrable. Additionally, both samplings generate uneven isocontours and particles sit at a significant offset from the actual boundary geometry.

For a 1:1 ratio (see Fig. 6 middle), our approach preserves the vertical sections of the boundary geometry, for all isocontours, whereas the boundary-integral approach significantly smoothes out the boundary geometry. Our approach preserves the geometry as, by construction, the isocontour for a particle of density 0.5 is coincidental with the boundary geometry, which ensures that the transition from outside to inside the boundary is always centered



(a) 1 : 4 ratio (b) 1 : 1 ratio (c) 4 : 1 ratio

Fig. 6. A numerical evaluation of how different boundary handling schemes can resolve a fixed size object (red, 1×1 unit) with different ratios of support radius to object size. For each variant the top row shows a particle-based method with a coarse (left) and fine (right) sampling and the bottom row shows a numerical boundary integral method (left) and our SDF-based method (right). The boundary contribution term λ is color coded from 0 (purple) to 1 (yellow). The isocontours show where a lone particle (black) and particles with quarter (blue), half (tan) and three quarter (purple) full neighborhoods would rest.

around the actual geometry. The particle-based approach is impenetrable and yields similar results for both sampling densities, but also smoothens out the boundary geometry and still exhibits uneven isocontours.

For a 4:1 ratio (see Fig. 6 right), both the boundary-integral approach and the particle-based approach smooth out the feature extremely, while our approach still resolves the boundary geometry properly for particles with at least a quarter full neighborhood.

9.3 Penalty Functions

There still remains a problem of an imprecise recovery of the location of isolated particles for high particle-to-feature size ratios (see the black isocontour for the 4:1 ratio, Fig 6 top rightmost). As our approach centers the boundary transition on the surface, the distance at which particles, with certain densities, rest is constant with respect to the surface of the boundary. Accordingly, any boundary feature smaller than twice this separation distance for isolated particles can be penetrated. This issue can be addressed by the idea of penalty functions, which modify the contribution of the boundary based on the distance to the boundary; see Sec. 5.5. Geometrically speaking, penalty functions shift and shrink the isocontours relative to the boundary geometry.

We now compare our approach with no penalty function, with the linear penalty function of Koschier and Bender [2017], the cubic penalty function of Bender et al. [2019] and with a modified softmax function $\beta(d) = \frac{\ln(1+e^{-2.5d})-\ln(1+e^{-2.5})}{\ln(2)}$; see Fig. 7. The cubic penalty function is equal to 1 on the boundary interior and, as such, fails to resolve the penetration issue and instead only compresses isocontours on the outside. The linear penalty function, and the softmax function, achieve an isocontour for isolated particles that follows the feature outline. As previously discussed by Bender et al. [2019], the linear penalty function is not differentiable at distance h from the boundary, which might cause undesirable behaviour in a simulation. In general, the softmax penalty function, which is continuously differentiable for any distance, allows for the interaction of particles with very small obstacles, i.e., with features that are 20 times smaller than the support radius. In practice, we utilize the linear penalty function, as it allows for the interaction with small boundary features, but does not incur the same additional computational cost.

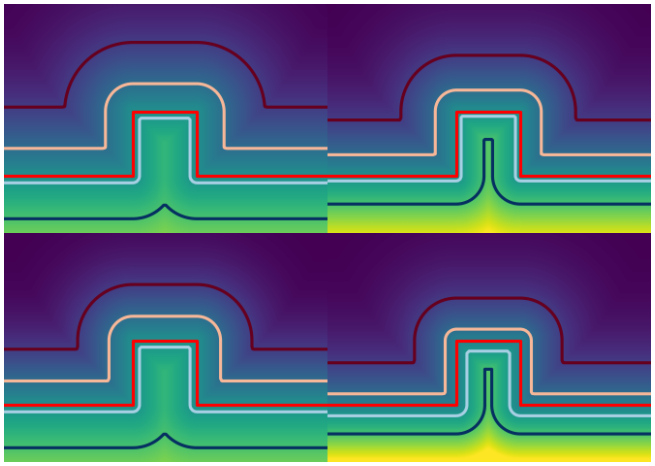


Fig. 7. Comparison of different penalty functions for the 4:1 ratio of Fig. 6. Here, the top left uses no penalty function, top right uses the penalty function of Koschier and Bender [2017], bottom left uses the penalty function of Bender et al. [2019], and bottom right uses a modified softmax function. λ color coded from 0 (purple) to 1 (yellow).The penalty functions significantly impact the representaiton of a small feature, especially for lone particles (black isocontour), i.e., the obstacle might only be repsented as a bump.

9.4 Simulating Small Boundary Features in 2D

To evaluate the impact of the difference in boundary representation further, we compare the difference of our approach and the numerical boundary-integral approach in a 2D dambreak scenario with an obstacle of fixed size (1×1 unit) and particles with relative support radii of 2, 1, 1/2 and 1/4; see Fig. 8. Considering our approach (top row), we can observe a similar deflection angle and overall behavior regardless of particle resolution, barring a variation of about 2 particle radii, with no penetration of the boundary object at any resolution. However, our method, at any resolution, has a single particle sitting on the bottom left vertex of the obstacle, which will be further investigated later. Considering the boundary-integral approach (Fig. 8 bottom row), we do not observe a consistent deflection angle and significant penetration of the boundary feature, especially at the 2:1 ratio. Moreover, the deflected flow becomes more collimated and the deflection angle gets steeper for finer particle resolutions due to the boundary becoming less smoothed out; see also Sec. 9.2. Only for even higher particle resolutions, i.e., 1/64th of the feature size, the deflection angle converges and coincides with the one in our approach. However, these high resolutions require inappropriatly high computational resources. Our approach preserves the overall behavior and yields very consistent deflection angles across a wide range of particle resolutions and, thus, resolves significantly smaller boundary geometry than prior methods.

To investigate this effect we evaluate the influence of the boundary sharpness by varying the angle of the boundary in a corner scene from acute to obtuse; see Fig. 9. In all three cases our approach yields nearly identical fluid behavior on the fluid surface. Near to the corner, however, as the corner angle becomes more acute, a single particle becomes trapped close to the corner. Whilst this effect does not introduce artifacts into the overall flow behavior, it can be visually apparent. We, therefore, avoid this effect on orthogonal simulation domain boundaries by rounding off the corners.

9.5 Discretized Signed Distance Fields

While a continuous SDF can represent arbitrary geometry exactly, discretizing the SDF commonly yields a loss of fine details and may introduce sampling artifacts,e.g., aliasing effects. We chose to represent SDFs as a hierarchical sparse voxel structure using OpenVDB [Museth et al. 2013] and GVDB [Hoetzlein 2016], where the resolution of the SDF is controlled by the voxel size. To determine the ideal voxel size to represent a given boundary mesh, we performed the process shown in Algorithm 3 to determine the best voxel size iteratively. In this process a candidate SDF is generated for a given voxel size and the mesh distance of a mesh reconstructed from this SDF and the initial boundary mesh is computed. Based on this mesh distance we then either further reduce the voxel size or stop the process if a certain error threshold is met.

As the error threshold we evaluated the maximum mesh distance and checked if this distance is less than the smallest particle radius expected within the simulations. This threshold results in a ratio of support radius, determined as $h_i = \sqrt[3]{50} r_i \approx 3.68 r_i$ [Winchenbach et al. 2016], to boundary accuracy of at least $3.68 : 1$. For the propeller scenario (Fig. 1) the maximum mesh distance was $4.45 \cdot 10^{-2}$, with a mean distance of $2.7 \cdot 10^{-4}$, for a finest particle radius of $6.46 \cdot 10^{-2}$, with an according smallest support radius of $2.38 \cdot 10^{-1}$,

ALGORITHM 3: Given an input boundary mesh $\mathcal{M}$, this procedure finds the appropriate discrete SDF $\mathcal{B}$ that represents the input boundary geometry to within an error threshold η.

$i \leftarrow 0$ **Do**
  $r \leftarrow r_{\text{base}} \left(\frac{1}{2}\right)^i ; i \leftarrow i + 1$
  Generate candidate SDF $\mathcal{B}'$ with resolution r
  Reconstruct the isosurface of $\mathcal{B}'$, yielding a mesh $\mathcal{M}'$
  Evaluate the mesh difference $D = \text{dist}(\mathcal{M}, \mathcal{M}')$
  $d \leftarrow \sup_{d' \in D} d'$
while d > η
Return SDF $\mathcal{B}'$

resulting in a maximum ratio of, approximately, $5.35 : 1$. Note that a lower voxel resolution could be utilized if the maximum error is constrained to spatial regions that are not important, e.g., if no fluid particles reach these regions, as the mean error is usually significant lower than the maximum. Furthermore, adaptive SDF representations, e.g., [Koschier et al. 2016], could also be utilized to adjust the error locally based on the complexity of the geometry, but in our scenarios this was not necessary.

9.6 Comparison of Boundary Handling in 3D

We first compare our method with Interlinked SPH [Gissler et al. 2019] in the one-way coupled *Dragon* scene, where a fluid volume impacts a static Stanford dragon model; see Fig. 10. Here, we observe an overall speedup of 1.5× times for our approach when compared to ILSPH; see Tab. 1. In more detail, the lower overall particle count (no boundary particles) reduces the cost of, for example, the neighborlist creation and the moving-least-squares pressure extrapolation reduces the overall iteration count, however each individual iteration is computationally slightly more expensive. Moreover, the simulation using the volume maps approach [Bender et al. 2019] also shows a higher overall iteration count, due to the lack of a pressure extrapolation approach. However, note that the overall timings are not comparable, as the volume maps approach is simulated on a CPU instead of on a GPU. Furthermore, there are significant visual differences between the three simulation approaches. The main difference is due to the fluid not flowing through some openings of the dragon; see the close-ups in Fig. 10. For the volume maps approach this issue arises due to a strong dependence on the sampling resolution of the boundary geometry, which in this case was chosen equal to the sampling resolution used in [Bender et al. 2019] for the dragon model. Increasing the sampling resolution, i.e., by a factor of 16 in each direction, can reduce some of these sampling deficiencies but requires approximately 2 hours for precomputing the volume map and over 50GB of memory to store it, making it difficult to use, especially on GPUs. Furthermore, the volume, and density, map construction processes also involve first evaluating an SDF that represents the boundary geometry and, accordingly, are also affected from issues inherent to SDFs. Considering the particle-based approach, the difference results from the offset of the boundary geometry for the particle-based representation that implicitly narrows small object cavities and holes and blocks fluid particles from entering these regions.

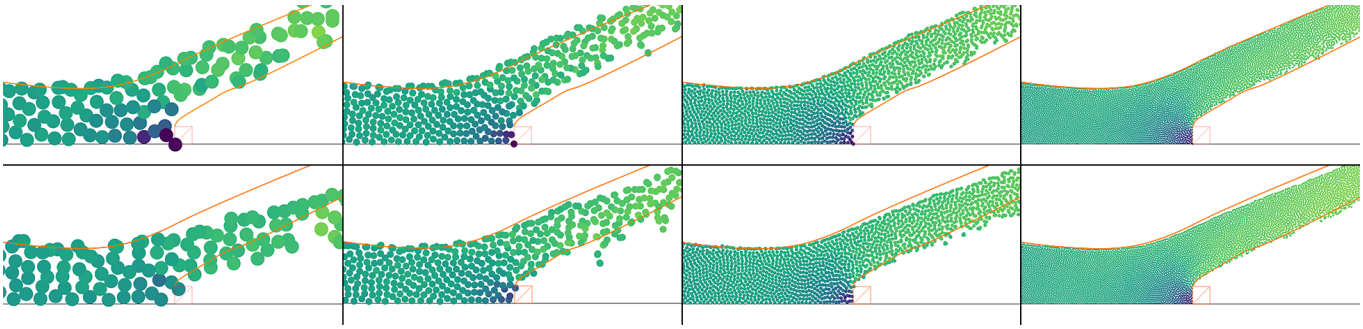


Fig. 8. Comparison of our semi-analytical approach (top row) to a numerical boundary-integral approach (bottom row) for an obstacle of fixed size (1×1 units) and particles with support radius 2, 1, 1/2 and 1/4 (from left to right). The orange outline superimposes the outline of the plume of the top right variant to all other variants. Velocity color coded from purple ($0m/s$) to yellow ($25m/s$).

Table 1. Quantitative comparison. All performance numbers are average values with respect to 30 seconds simulation time and timings refer 1/60s of simulation time. The radius values refers to the largest fluid particle radius in the simulation, and the ratio refers to the volume ratio of the largest to smallest particle volume. Incomp. and Div.-Free refer to the incompressible and divergence-free solver, respectively. In all results the divergence-free solver required 2 iterations.

Scene	**Figure**	n_{fluid}	n_{rigid}	**radius** /m	**ratio**	**timestep** /ms	**Incomp.** Iterations	**Frame-time** /ms	**Div.-Free** /ms	**Incomp.** /ms	**Adapt** /ms
Basin	14b	23K		0.5		8.0	2.86	60	5	13	
Basin	14c	23K	81K	0.5		8.0	6.48	74	4	8	
Basin	14d	3.17M		0.1		2.0	4.6	2661	262	1218	
Basin	14e	3.17M	2.05M	0.1		1.9	9.5	5231	392	2263	
Dragon	10 left	133K	183K	0.5		8.0	12.58	146	14	53	
Dragon	10 middle	133K		0.5		7.5	15.87	384	32	225	
Dragon	10 right	133K		0.5		8.0	10.71	94	7	53	
Dragon	11	1.98M		0.5	1000	3.9	7.2	2301	105	310	422
Propeller		539K		0.3		3.4	41.5	1642	40	1081	
Propeller	1	1.98M		0.3	100	0.7	3.1	9501	347	871	1321
Incline	17 left	56K		0.5		8.0	8.0	112	7	35	
Incline	17 right	56K		0.5		8.0	8.1	111	6	35	
Boxes	12	196K		0.5		6.0	9.3	267	12	113	
Boxes	13	1.65M		0.5	500	3.3	7.5	3561	116	376	450
Boxes		196K	124K	0.5		6.0	24.3	532	11	367	

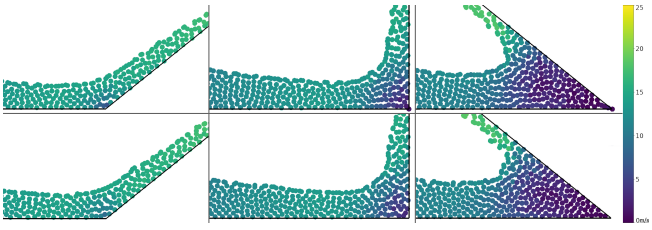
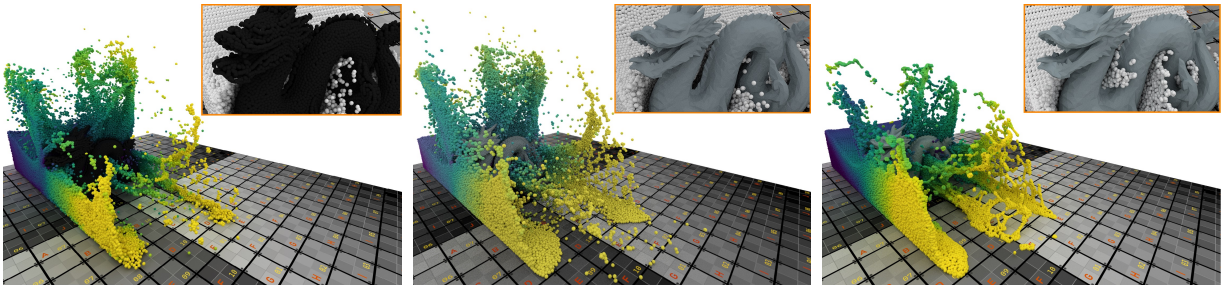


Fig. 9. Comparison of our semi-analytical approach (top row) to a numerical approach (bottom row) for an obtuse (left), an orthogonal (middle) and an acute angle (right) in 2D. Our semi-analytical solution provides very similar overall flow behavior, except for a single particle resting on the actual corner. Velocity color coded from purple ($0m/s$) to yellow ($25m/s$).

This is in accordance with Gissler et al. [2019] who observed a similar behavior considering rigid-rigid separation. They adjusted the separation distance by tuning the parameter γ in (12) in [Band et al. 2018a], which is not possible for fluid-rigid interactions as this could lead to holes in the boundary geometry; see Fig. 6. Fig. 12 shows a two-way coupled *Boxes* scene containing two-way coupled boxes being impacted by a fluid volume. We achieve a speedup of 2×, as the required iteration count is significantly reduced, which is in accordance with the results for one-way coupling and [Band et al. 2018a]. The difference in boundary representation causes significant differences between the overall flow behaviors; see the results for one-way coupling and Sec. 9.2. This difference is due to the feedback between fluid and boundary for two-way coupling, i.e., the fluid's dynamic and the boxes' motions are both influenced by the smoothed boundary-representation in a feedback loop. However,



(a) ILSPH [Gissler et al. 2019] (b) Volume Maps [Bender et al. 2019] (c) Our approach

Fig. 10. Simulation of a fluid volume hitting a complex boundary object (Stanford dragon) with different boundary handling approaches. Velocity color coded from purple ($0m/s$) to yellow ($30m/s$) for the main view. The closeups show an earlier timepoint with fluid particles as white and boundary particles as black.

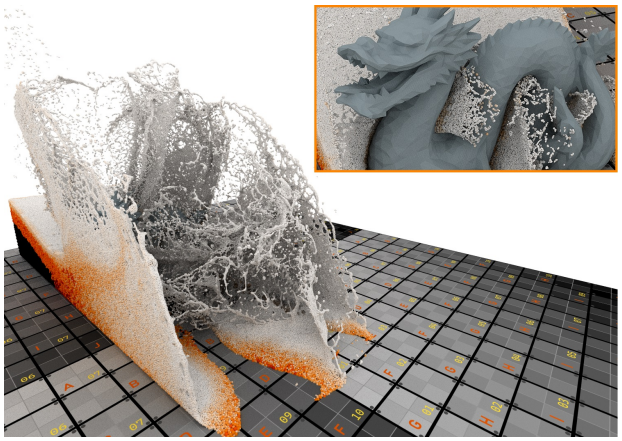


Fig. 11. Adaptive variant of the *Dragon* scene using our boundary handling approach and particle volume color coded from black (high) to white (low).

these issues are just an amplification of a problem already discussed in context of Fig. 10.

Note that whilst our single contact point model can readily handle two-way fluid-rigid coupling and provides a significant speedup over particle-based methods, simulating strong fluid-rigid coupling, as done by Gissler et al. [2019], is not possible as there is no way to evaluate boundary-boundary interactions in our model and, thus, have to rely on external rigid-rigid solvers, i.e., Bullet [Coumans 2015].

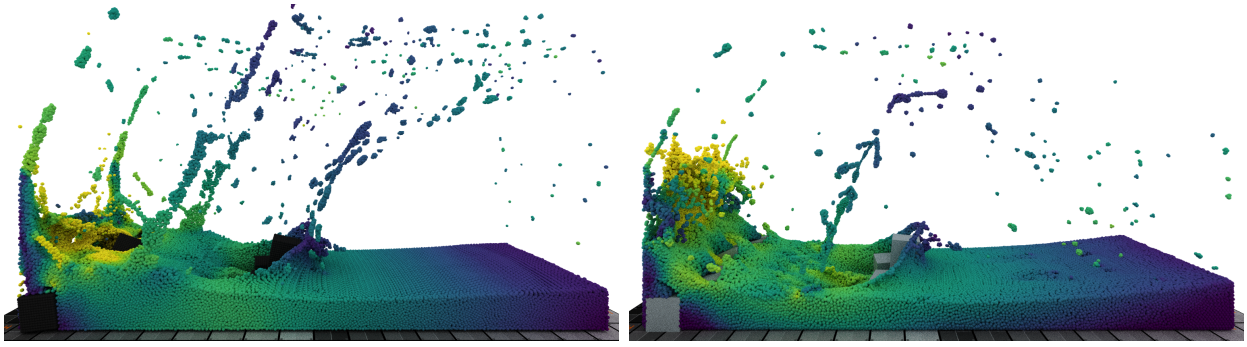
9.7 Sampling

Fig. 14 compares different fluid particle resolutions and different boundary representations in a simple scenario where fluid is dropped into a basin. We use two different fluid particle resolutions, $r = 0.5$ and $r = 0.075$, and a fixed voxel resolution of 0.5 for the SDF. For the resolution of $r = 0.5$ and $r = 0.075$, the boundary particle sampling requires about 15 seconds and 81K boundary particles and almost fifteen minutes and 2M boundary particles, respectively. At matching fluid and boundary particle resolutions the simulation is stable, i.e., no particles explode in the simulation, but significant problems are visible where particles get stuck on the surface due to the uneven sampling caused by particles, also observed by Band et al. [2017]. This sampling introduces noticeable artificial viscosity, which causes the overall fluid behavior to change. In Fig. 10 the fluid front for the particle-based approach lages behind the SDF-based boundary with equal fluid resolution. Mismatching resolutions either result in unstable simulations, in case boundary particles are too small, and penetration in case of boundary particles too large; see also Sec. 9.2. The boundary-integral-based nature of our approach intrinsically solves this sampling problem and yields stable simulations, as the SDF results in a smooth and continous boundary representation that is independent from the fluid particle resolution. Fig. 15 nicely demonstrates the flexibility of SDFs for very large scenes with small boundary features.

9.8 Adaptivity

In this section we briefly summarize the behavior of our boundary handling method related to spatially adaptive fluids. We observe no instabilities directly caused by the interaction of particles of varying resolutions with the same boundary geometry. Moreover, we observe similar flows in adaptive and non-adaptive flows close to boundaries, e.g., Fig 12 and Fig. 13, due the underlying consistent behavior of our method across varying resolutions; see Sec. 9.2. Fig. 16 demonstrates this for smoothly varying resolutions along the boundary of a one-way coupled scenario. Regarding computational cost, our boundary handling approach has no significant drawback for an adaptive simulation, i.e., it does not require sampling the boundary with boundary particles of the finest possible fluid particle resolution. However, note that we cannot evaluate the computational cost for a baseline resolution, i.e., all particles are at the finest resolution of the adaptive simulation, as this would require, for example, 133 million particles in the *Dragon* scene. This is significantly above the maximum number of 23 million particles that we can simulate on our hardware, using the data structure approach of Winchenbach and Kolb [2019].

Fast moving objects, i.e., in the *Propeller* scene in Fig. 1, require a significantly reduced timestep, but this is due to the requirements imposed by the CFL condition and not specifically due to our boundary handling scheme. Furthermore, for the adaptive method of Winchenbach et al. [2017] the particle resolution depends on the distance of



(a) ILSPH [Gissler et al. 2019] (b) Our approach, uniform resolution

Fig. 12. Comparison for the *Boxes* scenario, using different boundary handling approaches, where a set of stacked boxes is impacted by a fluid volume. Due to the different representation of the boundaries the results are noticeably different, i.e., the orientation of the box in the bottom left corner as well as the exaggerated splashing due to the representation of the narrow passages between boxes on the initial impact. Velocity color coded from purple to yellow.

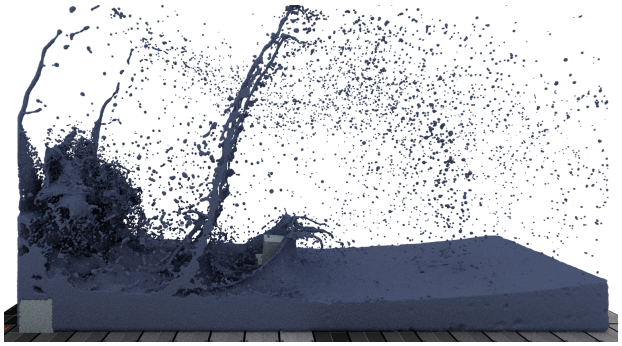


Fig. 13. Spatially adaptive variant of Fig. 12 rendered with an extracted surface using [Zhu and Bridson 2005]. Adaptivity causes changing resolutions on the outside walls, which causes artifacts during surface extraction.

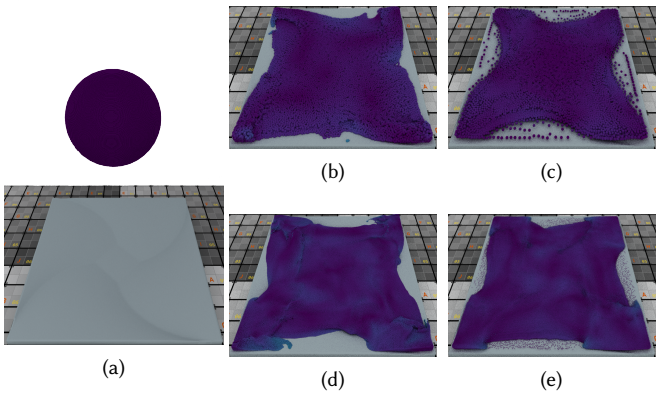


Fig. 14. Comparison of our semi-analytical boundary handling approach (middle) and a particle-based boundary handling approach (right) considering the behavior of a boundary at a fixed sampling with low (top) and high (bottom) resolution particles. The volume ratio between the low and high fluid particle resolution was about 300 : 1 with a boundary particle resolution equal to the fluid particle resolution. (a) shows the initial setup of fluid before dropping. Velocity of particles is color coded, with boundary particles being colored dark grey.

a particle to the fluid surface. Existing surface detection methods can sometimes yield incorrect results, i.e., for thin fluid sheets or lone clusters of particles causing particles to needlessly change resolution, which can introduce visual artifacts, that are amplified by parameter scaling problems in surface extraction methods.

9.9 Friction

In order to evaluate the friction model presented in Sec. 8, we drop a sphere of liquid onto an inclined plane with a high friction coefficient in one case and zero friction in the other case; see Fig. 17. This demonstrates the capability of our method to simulate boundaries ranging from free-slip to no-slip boundary conditions. Even though the Coulomb friction model should, from a physical perspective, not be applied to fluid-solid interactions, it still yields visually pleasing results. It has been demonstrated in previous work that a particle-based boundary representation leads to undesirable results due to sampling irregularities [Koschier and Bender 2017].

9.10 Stability

The stability of our method is evaluated in a complex one-way coupled simulation, shown in Fig. 1. Here an inlet stream from the right (initial velocity of $30m/s$) collides with a counter clockwise rotating propeller in a closed housing. Due to the limited space between the housing and the propeller, and due to the propeller pushing the fluid back towards the inlet stream, a significant compressive stress is generated. For a uniform particle resolution, as given in Fig. 1, the average timestep requires 42 iterations of the incompressible solver at a CFL limited timestep of 3.4ms. However, the simulation maintains an average compression error of less than 0.01%. In general, high compressive stresses are in regions of low resolution in the fluid bulk, and not at the free surface with higher particle resolution.

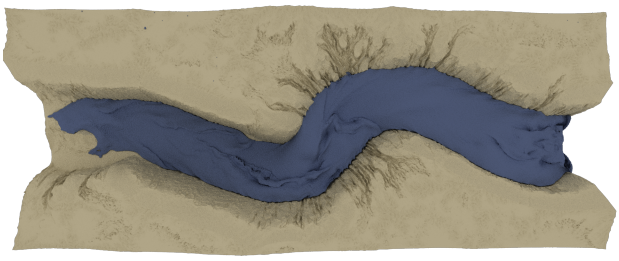


Fig. 15. This figure shows a *River* scene where a fluid inlet stream, at a particle radius of $0.5m$ moves through a canyon that is 1km long, 400m wide and 200m high. In this scene up to 8 million particles are simulated. The surface is extracted using the approach of Zhu and Bridson [2005].

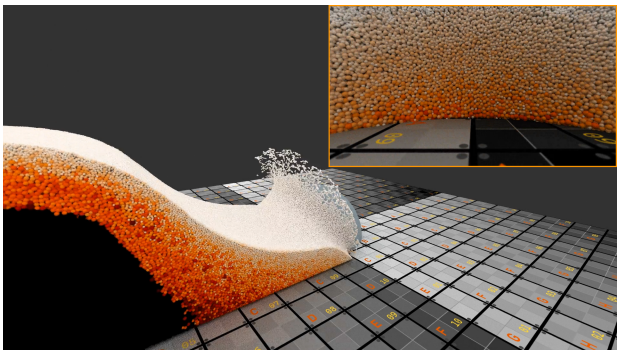


Fig. 16. An adaptive dambreak simulation where an initial fluid volume impacts a smooth hemisphere. The top right close-up shows a view from inside the hemisphere to demonstrate the changing resolution along the boundary. Particle volume color coded.

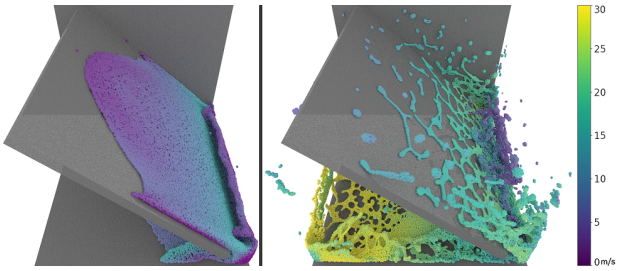


Fig. 17. Simulation of the collision of an initial spherical fluid volume falling on an inclined surface. On the left we set the friction coefficient μ for all boundaries equal to 1, whereas the right uses a friction coefficient $\mu = 0$. Our method is readily capable of simulating a wide range of fluid-rigid interactions, ranging from free-slip (right) to no-slip boundary conditions. Velocity color coded from purple ($0m/s$) to yellow ($30m/s$).

Consequently, when using an adaptive particle resolution in this scenario (Fig. 1), the average timestep is reduced to $740\mu s$ as the smallest particle has a 4.6 times smaller radius. This, in turn, reduces the timestep by a factor of 4.6 due to the CFL condition. As a consequence, the average number of iterations of the pressure solver is reduced to 3, which significantly reduces the computational requirements per particle. All in all, 22 timesteps are required per rendered frame, which significantly slowed down the overall simulation.

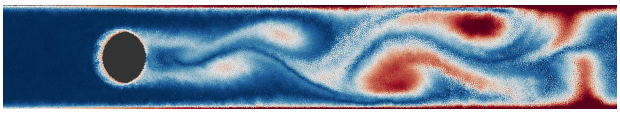


Fig. 18. This figure demonstrates a flow past a cylinder simulation in 3D for an inviscid fluid demonstrating vortex shedding behind the obstacle. Local angular velocity is color coded from zero (blue) to high (red). Note that particles close to the boundary have a relatively high angular velocity as the boundary is not moving relative to them.

To further evaluate the stability of our method, we simulate a fluid stream flowing past a cylinder through a tube in 3D; see Fig. 18. For this simulation we include the background pressure term of Marrone et al. [2013], to avoid the formation of voids behind the cylinder. Our boundary handling approach is able to simulate this scenario, including vortex shedding behind the cylinder, even for inviscid fluids at high velocities. Due to the smooth nature of the boundaries in this scene our method closely approximates the results of a boundary-integral approach.

9.11 Limitations

Our method has three major limitations. Firstly, SDFs can not represent all types of geometries consistently. In regions where multiple parts of a single rigid body are within the support radius of a particle, e.g., in narrow cavities, the single contact point model might not accurately represent the interaction. However, this effect can be reduced by performing a convex body decomposition, apriori, yielding multiple contact points. Secondly, single particles get trapped in sharp boundary features. Even though this does not negatively impact the flow behavior, this is an undesired property which we intend to address using improved penalty functions in future work. Finally, our approach cannot readily handle deformable boundary geometries, without significant computational overhead. Additionally, certain aspects of SPH, in general, are not yet well applied to spatially adaptive methods, i.e., surface-detection and surface-extraction methods, inducing visual artifacts. Moreover, certain aspects of spatially adaptive SPH methods, i.e., how to merge large amounts of high-resolution particles, are challenging problems, in general, and can potentially cause simulations to not behave well.

10 CONCLUSIONS

In this paper we presented a novel semi-analytic boundary handling approach for SPH simulations. The core idea is to define a locally representative planar boundary, for arbitrary boundary geometries, using an SDF, and to derive an analytic solution for the interaction with planar boundaries to this representative boundary. Our method yields significantly improved preservation of detail for small boundary features and provides consistent boundary interactions across varying particle resolutions. Additionally, our approach can readily be integrated into existing SPH simulation methods and can handle spatially adaptive simulations, even in two-way coupled scenarios.

REFERENCES

Stefan Adami, Xiangyu Y Hu, and Nikolaus A Adams. 2012. A generalized wall boundary condition for smoothed particle hydrodynamics. *J. Comput. Phys.* 231, 21 (2012), 7057–7075.

Bart Adams, Mark Pauly, Richard Keiser, and Leonidas J Guibas. 2007. Adaptively sampled particle fluids. In *ACM Transactions on Graphics (TOG)*, Vol. 26. Acm, 48.

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

Nadir Akinci, Jens Cornelis, Gizem Akinci, and Matthias Teschner. 2013b. 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), 62.

Stefan Band, Christoph Gissler, Markus Ihmsen, Jens Cornelis, Andreas Peer, and Matthias Teschner. 2018a. Pressure boundaries for implicit incompressible SPH. *ACM Transactions on Graphics (TOG)* 37, 2 (2018), 14.

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

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.

Markus Becker, Hendrik Tessendorf, and Matthias Teschner. 2009. Direct forcing for lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics* 15, 3 (2009), 493–503.

Nathan Bell, Yizhou Yu, and Peter J Mucha. 2005. Particle-based simulation of granular materials. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. ACM, 77–86.

Jan Bender. 2016. *SPlisHSPlasH*. http://www.interactive-graphics.de/index.php/downloads/106-splishsplash-library Accessed: 2020-08-04.

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

Jan Bender, Tassilo Kugelstadt, Marcel Weiler, and Dan Koschier. 2019. Volume Maps: An Implicit Boundary Representation for SPH. In *Motion, Interaction and Games*. ACM, 26.

L Chiron, Matthieu De Leffe, Guillaume Oger, and David Le Touzé. 2019. Fast and accurate SPH modelling of 3D complex wall boundaries in viscous and non viscous flows. *Computer Physics Communications* 234 (2019), 93–111.

Erwin Coumans. 2015. Bullet Physics Simulation. In *ACM SIGGRAPH 2015 Courses (SIGGRAPH '15)*. ACM, New York, NY, USA, Article 7. https://doi.org/10.1145/2776880.2792704

Walter Dehnen and Hossam Aly. 2012. Improving convergence in smoothed particle hydrodynamics simulations without pairing instability. *Monthly Notices of the Royal Astronomical Society* 425, 2 (2012), 1068–1082.

Jonathan A. Feldman and Javier Bonet. 2007. Dynamic refinement and boundary contact forces in SPH with applications in fluid flow problems. *Internat. J. Numer. Methods Engrg.* 72, 3 (2007), 295–324.

Martin Ferrand, DR Laurence, Benedict D Rogers, Damien Violeau, and Christophe Kassiotis. 2013. Unified semi-analytical wall boundary conditions for inviscid, laminar or turbulent flows in the meshless SPH method. *International Journal for Numerical Methods in Fluids* 71, 4 (2013), 446–472.

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.

R. A. Gingold and J. J. Monaghan. 1977. Smoothed particle hydrodynamics-theory and application to non-spherical stars. *Monthly Notices of the Roy. Astronomical Soc.* 181 (1977), 375–389.

Christoph Gissler, Stefan Band, Andreas Peer, Markus Ihmsen, and Matthias Teschner. 2017. Generalized drag force for particle-based simulations. *Computers & Graphics* 69 (2017), 1–11.

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), 5.

Rama Karl Hoetzlein. 2016. GVDB: Raytracing Sparse Voxel Database Structures on the GPU. In *Proceedings of High Performance Graphics (HPG '16)*. Eurographics Association, Goslar Germany, Germany, 109–117. https://doi.org/10.2312/hpg.20161197

Christopher Jon Horvath and Barbara Solenthaler. 2013. *Mass Preserving Multi-Scale SPH*. Technical Report. Emeryville, CA.

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.

Markus Ihmsen, Jens Orthmann, Barbara Solenthaler, Andreas Kolb, and Matthias Teschner. 2014. SPH Fluids in Computer Graphics. *Eurographics STARS* 2 (2014), 21–42.

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

Dan Koschier, Jan Bender, Barbara Solenthaler, and Matthias Teschner. 2019. Smoothed Particle Hydrodynamics Techniques for the Physics Based Simulation of Fluids and Solids. In *Eurographics 2019 - Tutorials*, Wenzel Jakob and Enrico Puppo (Eds.). The Eurographics Association, 1–41. https://doi.org/10.2312/egt.20191035

Dan Koschier, Crispin Deul, and Jan Bender. 2016. Hierarchical hp-adaptive signed distance fields.. In *Symposium on Computer Animation*. 189–198.

Agnès Leroy, Damien Violeau, Martin Ferrand, and Christophe Kassiotis. 2014. Unified semi-analytical wall boundary conditions applied to 2-D incompressible SPH. *J. Comput. Phys.* 261 (2014), 106–129.

Frank Losasso, Jerry Talton, Nipun Kwatra, and Ronald Fedkiw. 2008. Two-way coupled SPH and particle level set fluid simulation. *IEEE Transactions on Visualization and Computer Graphics* 14, 4 (2008), 797–804.

Miles Macklin and Matthias Müller. 2013. Position based fluids. *ACM Trans. Graph.* 32, 4 (2013), 1. https://doi.org/10.1145/2461912.2461984

Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. 2014. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 153.

Salvatore Marrone, Andrea Colagrossi, Matteo Antuono, G Colicchio, and Giorgio Graziani. 2013. An accurate SPH modeling of viscous flows around bodies at low and moderate Reynolds numbers. *J. Comput. Phys.* 245 (2013), 456–475.

Arno Mayrhofer, Martin Ferrand, Christophe Kassiotis, Damien Violeau, and François-Xavier Morel. 2015. Unified semi-analytical wall boundary conditions in SPH: analytical extension to 3-D. *Numerical Algorithms* 68, 1 (2015), 15–34.

Aleka McAdams, Andrew Selle, Rasmus Tamstorf, Joseph Teran, and Eftychios Sifakis. 2011. *Computing the singular value decomposition of 3x3 matrices with minimal branching and elementary floating point operations*. Technical Report. University of Wisconsin-Madison Department of Computer Sciences.

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

Joseph J Monaghan. 2002. SPH compressible turbulence. *Monthly Notices of the Royal Astronomical Society* 335, 3 (2002), 843–852.

Joseph J Monaghan. 2005. Smoothed particle hydrodynamics. *Reports on progress in physics* 68, 8 (2005), 1703.

Matthias Müller, David Charypar, and Markus Gross. 2003. Particle-Based Fluid Simulation for Interactive Applications. *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animat.* 5 (2003), 154–159.

Ken Museth, Jeff Lait, John Johanson, Jeff Budsberg, Ron Henderson, Mihai Alden, Peter Cucka, David Hill, and Andrew Pearce. 2013. OpenVDB: an open-source data structure and toolkit for high-resolution volumes. In *Acm siggraph 2013 courses*. ACM, 19.

Daniel J Price. 2012. Smoothed particle hydrodynamics and magnetohydrodynamics. *J. Comput. Phys.* 231, 3 (2012), 759–794.

Benedict D Rogers, Robert A Dalrymple, and Peter K Stansby. 2010. Simulation of caisson breakwater movement using 2-D SPH. *Journal of Hydraulic Research* 48, S1 (2010), 135–141.

Barbara Solenthaler and Markus Gross. 2011. Two-scale particle simulation. In *ACM Transactions on Graphics (TOG)*, Vol. 30. ACM, 81.

Barbara Solenthaler and Renato Pajarola. 2009. Predictive-corrective incompressible SPH. In *ACM transactions on graphics (TOG)*, Vol. 28. ACM, 40.

Rene Winchenbach. 2018. *Fast 3x3 SVDs for GPUs and CPUs*. https://github.com/wire/tbtSVD Accessed: 2020-08-04.

Rene Winchenbach. 2019. *openMaelstrom*. http://www.cg.informatik.uni-siegen.de/openMaelstrom Accessed: 2020-08-04.

Rene Winchenbach, Hendrik Hochstetter, and Andreas Kolb. 2016. Constrained Neighbor Lists for SPH-based Fluid Simulations. In *Proceedings of the 15th ACM SIGGRAPH/Eurographics symposium on computer animation*.

Rene Winchenbach, Hendrik Hochstetter, and Andreas Kolb. 2017. Infinite Continuous Adaptivity for Incompressible SPH. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 102:1–102:10.

Rene Winchenbach and Andreas Kolb. 2019. Multi-Level-Memory Structures for Adaptive SPH Simulations. (2019).

Hong Xiao and Zydrunas Gimbutas. 2010. A numerical algorithm for the construction of efficient quadrature rules in two and higher dimensions. *Computers & mathematics with applications* 59, 2 (2010), 663–676.

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

APPENDIX

The analytical solution in 2D required solving a set of definite integrals:

$$\begin{aligned}
\lambda_2(d) &= \begin{cases} \int_d^{0.5} f(q,d)dq + \int_{0.5}^1 g(q,d)dq, & 0 \le d \le 0.5 \\ \int_d^1 g(q,d)dq, & 0.5 < d \le 1, \\ 1 - \lambda_2(-d), & -1 \le d < 0 \end{cases} \\
f(q,d) &= 2\operatorname{acos}\left(\frac{d}{q}\right) qC_2 \left[(1-q)^3 - 4\left(\frac{1}{2}-q\right)^3\right], \\
g(q,d) &= 2\operatorname{acos}\left(\frac{d}{q}\right) qC_2 \left(1-q\right)^3,
\end{aligned} \tag{56}$$

which can be evaluated symbolically. However, the resulting intermediate definite integrals are skipped for brevity. The final boundary contribution term $\lambda_2(d)$ in 2D can be determined using the definite integrals from before as

$$\lambda_2(d) = -\frac{1}{7\pi} \begin{cases} d_1 + d_2 + d_3 + d_4 + d_5 + d_6, & 0 \le d < 0.5 \\ e_1 + e_2 + e_3 + e_4, & 0.5 \le d \le 1 \\ 7\pi\mathcal{W}(-d) - 7\pi, & -1 \le d < 0, \end{cases} \tag{57}$$

with

$$\begin{aligned}
d_1 &= \left(-12d^5 - 80d^3\right)\log\left(2\sqrt{1-d^2}+2\right) \\
d_2 &= \left(30d^5 + 80d^3\right)\log\left(\sqrt{1-4d^2}+1\right) \\
d_3 &= -18d^5\log\left(1-\sqrt{1-4d^2}\right); d_4 = \operatorname{acos}(2d) - 8\operatorname{acos}(d) \\
d_5 &= \sqrt{1-d^2}\left(68d^3+32d\right); d_6 = \sqrt{1-4d^2}\left(-68d^3-8d\right) \\
e_1 &= \left(-6d^5 - 40d^3\right)\log\left(\sqrt{1-d^2}+1\right); e_2 = -8\operatorname{acos}(d) \\
e_3 &= \left(6d^5 + 40d^3\right)\log\left(1-\sqrt{1-d^2}\right); e_4 = \sqrt{1-d^2}\left(68d^3+32d\right)
\end{aligned} \tag{58}$$

The analytical solution in 3D required solving a set of definite integrals

$$\begin{aligned}
\lambda_3(d) &= \begin{cases} \int_d^{0.5} f(q,d)dq + \int_{0.5}^1 g(q,d)dq, & 0 \le d \le 0.5 \\ \int_d^1 g(q,d)dq, & 0.5 < d \le 1, \\ 1 - \lambda_3(-d), & -1 \le d < 0, \end{cases} \\
f(q,d) &= C_3\left[(1-q)^3 - 4\left(\frac{1}{2}-q\right)^3\right] 2\pi q(q-d), \\
g(q,d) &= C_3\left(1-q\right)^3 2\pi q(q-d),
\end{aligned} \tag{59}$$

which result in

$$\begin{aligned}
\int_d^{0.5} f(q,d)dq &= \frac{192d^6 - 288d^5 + 160d^3 - 66d + 19}{60}, \\
\int_{0.5}^1 f(q,d)dq &= -\frac{18d-11}{60}, \\
\int_d^1 g(q,d)dq &= -\frac{16d^6 - 72d^5 + 120d^4 - 80d^3 + 24d - 8}{15}.
\end{aligned} \tag{60}$$

In order to calculate $\mathcal{S}$ in 3D the following definite integrals

$$\begin{aligned}
\mathcal{S}_3(d) &= \begin{cases} \int_d^{0.5} f(q,d)dq + \int_{0.5}^1 g(q,d)dq, & 0 \le d \le 0.5 \\ \int_d^1 g(q,d)dq, & 0.5 < d \le 1, \\ \int_0^{0.5} f(q,d)dq + \int_{0.5}^1 g(q,d) - \mathcal{S}_3(-d), & -1 \le d < 0, \end{cases} \\
f(q,d) &= \frac{4608}{\pi}\left[q^3\left(q-d\right)\left(3q-2\right)^2\right], \\
g(q,d) &= \frac{4608}{\pi}\left[q\left(q-d\right)\left(q-1\right)^4\right],
\end{aligned} \tag{61}$$

were solved as

$$\begin{aligned}
&\int_0^{0.5} f(q,d)dq = \frac{f_1+f_2}{420\pi}, \int_d^{0.5} f(q,d)dq = -\frac{2337d-578}{420\pi}, \\
&\int_{0.5}^1 g(q,d)dq = -\frac{81d-46}{252\pi}, \int_d^1 g(q,d)dq = \frac{g_1+g_2}{63\pi}, \\
&f_1 = 26880d^9 - 69120d^8 + 46080d^7 + 21504d^6, \\
&f_2 = -32256d^5 + 8960d^3 - 2337d + 578, \\
&g_1 = 448d^9 - 3456d^8 + 11520d^7 - 21504d^6, \\
&g_2 = 24192d^5 - 16128d^4 + 5376d^3 - 576d + 128,
\end{aligned} \tag{62}$$

which results in a final boundary term in 3D as

$$\mathcal{S}_3(d) = \begin{cases} \frac{1}{h^{d+2}}\frac{1}{315\pi}\left[s_1+s_2\right], & 0 \le d \le 0.5 \\ \frac{1}{h^{d+2}}\frac{1}{63\pi}\left[t_1+t_2\right], & 0.5 < d \le 1, \\ -\frac{1}{630\pi}\left[5913d - 5392\right] - \mathcal{S}(-d), & -1 \le d < 0. \end{cases} \tag{63}$$

with

$$\begin{aligned}
s_1 &= 20160d^9 - 51840d^8 + 34560d^7, \\
s_2 &= +16128d^6 - 24192d^5 + 6720d^3 - 1854d + 491, \\
t_1 &= 448d^9 - 3456d^8 + 11520d^7, \\
t_2 &= -21504d^6 + 24192d^5 - 16128d^4 + 5376d^3 - 576d + 128
\end{aligned} \tag{64}$$

The naïve gradient term, as shown in Sec. 5.4, can also be found for particle-based terms, i.e., [Price 2012]

$$\nabla_i A_i = \sum_j \frac{m_j}{\rho_j} A_j \nabla_i W_{ij}, \tag{65}$$

which also is not exact for constant functions and also does not preserve symmetric interactions. For a particle-based representation a term that is exact for constant functions [Price 2012] can be found for boundary integral approaches as

$$\nabla A_i = \sum_j \frac{m_j}{\rho_j}(A_i - A_j)\nabla_i W_{ij} \rightarrow \frac{\rho_b^0}{\rho_b}\left(A_i - A_B\right)\nabla_i \lambda_n\left(d\right), \tag{66}$$

and one that preserves symmetric interactions as

$$\nabla A_i = \rho_i \sum_j m_j \left(\frac{A_i}{\rho_i^2} - \frac{A_j}{\rho_j^2}\right)\nabla_i W_{ij} \rightarrow \rho_i \rho_b^0 \left(\frac{A_i}{\rho_i^2} - \frac{A_B}{\rho_b^2}\right)\nabla_i \lambda_n\left(d\right), \tag{67}$$

utilizing the same derivation as Price [2012].