Pressure Boundaries for Implicit Incompressible SPH

STEFAN BAND, University of Freiburg, Germany
CHRISTOPH GISSLER, University of Freiburg, Germany and FIFTY2 Technology GmbH, Germany
MARKUS IHMSEN, FIFTY2 Technology GmbH, Germany
JENS CORNELIS, FIFTY2 Technology GmbH, Germany
ANDREAS PEER, University of Freiburg, Germany
MATTHIAS TESCHNER, University of Freiburg, Germany

Implicit incompressible SPH (IISPH) solves a pressure Poisson equation (PPE). While the solution of the PPE provides pressure at fluid samples, the embedded boundary handling does not compute pressure at boundary samples. Instead, IISPH uses various approximations to remedy this deficiency. In this paper, we illustrate the issues of these IISPH approximations. We particularly derive Pressure Boundaries, a novel boundary handling that overcomes previous IISPH issues by the computation of physically meaningful pressure values at boundary samples. This is basically achieved with an extended PPE. We provide a detailed description of the approach that focuses on additional technical challenges due to the incorporation of boundary samples into the PPE. We therefore use volume-centric SPH discretizations instead of typically used density-centric ones. We further analyze the properties of the proposed boundary handling and compare it to the previous IISPH boundary handling. In addition to the fact that the proposed boundary handling provides physically meaningful pressure and pressure gradients at boundary samples, we show further benefits such as reduced pressure oscillations, improved solver convergence and larger possible time steps. The memory footprint of fluid samples is reduced and performance gain factors of up to five compared to IISPH are presented.

CCS Concepts: • **Computing methodologies → Physical simulation**; *Massively parallel and high-performance simulations*;

Additional Key Words and Phrases: physically-based animation, fluid animation, Smoothed Particle Hydrodynamics, boundary handling

ACM Reference Format:
Stefan Band, Christoph Gissler, Markus Ihmsen, Jens Cornelis, Andreas Peer, and Matthias Teschner. 2018. Pressure Boundaries for Implicit Incompressible SPH. *ACM Trans. Graph.* 37, 2, Article 14 (February 2018), 12 pages. https://doi.org/10.1145/3180486

1 INTRODUCTION

Incompressible SPH (ISPH) has recently gained attention in the graphics community as a promising alternative to previous non-iterative, e.g. [Becker and Teschner 2007; Müller et al. 2003], or iterative state-equation solvers, e.g. [He et al. 2012; Shao and Lo 2003; Solenthaler and Pajarola 2009]. ISPH solves a pressure Poisson equation (PPE) to compute pressure values at fluid samples in a global way, e.g. [Cummins and Rudman 1999; Goswami et al. 2010; Ihmsen et al. 2014a; Takahashi et al. 2016]. As outlined in [Ihmsen et al. 2014b], various source terms can be used in the PPE that either encode the predicted density error or the predicted divergence of the velocity field. These formulations can also be combined in various ways. E.g., [Bender and Koschier 2017] indicates that a combination of two pressure solvers that resolve both, density deviations and the divergence of the velocity field, can improve the overall performance of the pressure computation. In this paper, we build upon implicit incompressible SPH (IISPH) with the density invariance condition as source term [Ihmsen et al. 2014a].

Related work. Representing solid boundaries with particles is popular in SPH fluids [Bender and Koschier 2017; Ihmsen et al. 2014a; Monaghan 2005; Takahashi et al. 2016], since particle-based representations are very flexible and can handle arbitrarily shaped geometries. E.g., in [Monaghan 2005], boundary particles exert penalty forces on the surrounding fluid samples as soon as they are within a certain distance. Penalty forces should prevent fluid samples from penetrating the boundary. However, small time steps are required to produce a smooth pressure field, since these forces also lead to large pressure variations within the fluid. In order to achieve larger time steps, the direct forcing method has been proposed in [Becker et al. 2009]. In this method, one- and two-way-coupled solid objects are handled by computing control forces and velocities with a predictor-corrector-scheme. But, due to an incomplete support domain, approximating field variables at boundaries is problematic with SPH. As a result, fluid samples tend to stick to the boundary if the direct forcing method or a distance-based penalty force is used. Ghost particles [Colagrossi and Landrini 2003; Schechter and Bridson 2012; Yildiz et al. 2009] are another technique to treat boundaries. For fluid samples that are located at a certain distance to the boundary, a ghost particle is generated, which has the same viscosity, mass, density and pressure as its associated fluid sample. For complex geometries, however, generating such ghost particles is challenging. In addition, the sampling of the boundary has a significant influence on the numerical stability and quality of the simulation. While simple objects can be easily represented by uniformly distributed samples, e.g. [Adami et al. 2012], complex objects can not. Based on the concept of the number density [Ott and Schnetter 2003; Solenthaler and Pajarola 2008], [Akinci et al. 2012] treats irregular samplings by computing volume contributions and by mirroring

Authors' addresses: Stefan Band, bands@informatik.uni-freiburg.de, University of Freiburg, Georges-Köhler-Allee 52, Freiburg, 79110, Germany; Christoph Gissler, gisslerc@informatik.uni-freiburg.de, University of Freiburg, School of Engineering, Freiburg, 79110, Germany, FIFTY2 Technology GmbH, Freiburg, 79100, Germany; Markus Ihmsen, ihmsen@fifty2.eu, FIFTY2 Technology GmbH, Freiburg, 79100, Germany; Jens Cornelis, cornelis@fifty2.eu, FIFTY2 Technology GmbH, Freiburg, 79100, Germany; Andreas Peer, peera@informatik.uni-freiburg.de, University of Freiburg, School of Engineering, Freiburg, 79110, Germany; Matthias Teschner, teschner@informatik.uni-freiburg.de, University of Freiburg, School of Engineering, Freiburg, 79110, Germany.

the hydrodynamic quantities of a fluid sample, i.e. density and pressure, onto its neighboring boundary samples. While adhering to the concept of SPH, this approach is efficient to compute and allows a versatile coupling of fluids and solid objects. Employing a Moving Least Squares technique, [Band et al. 2017] extended the boundary handling of [Akinci et al. 2012] to improve the accuracy of the density estimate and normal computation in planar regions. They try to locally reconstruct the surface of the true boundary by fitting boundary particles to a plane, resulting in a smooth representation of the boundary. Recently, an implicit representation of static and dynamic rigid boundaries has been proposed [Koschier and Bender 2017]. This approach allows an efficient density and boundary normal evaluation based on a pre-computed density map. All these techniques, however, do not solve a system to compute consistent pressure at the boundary.

Our contribution. We propose Pressure Boundaries, a novel boundary handling for IISPH that computes pressure values at boundary samples using the PPE. In contrast to IISPH, approximate or inconsistent assumptions are avoided. Instead, pressure values are computed that realize a physically meaningful pressure gradient at the discretized fluid-solid interface. This improves the robustness of the boundary handling, in particular the number of required solver iterations is reduced compared to IISPH. Performance gain factors of up to five have been achieved in the presented experiments. The proposed solution also works with larger time steps compared to IISPH. Further, Pressure Boundaries reduce artificial pressure oscillations that can occur in IISPH due to simplifying assumptions on boundary pressures.

Differences and benefits over IISPH. IISPH mirrors pressure values from fluid samples to adjacent boundary samples which has various issues. First, the pressure gradient is not correct at fluid-solid interfaces. Second, a boundary sample can have different pressure values mirrored from adjacent fluid samples. Third, all boundary samples adjacent to a fluid particle have the same pressure value. These issues are addressed in the proposed approach by computing unique, physically meaningful pressure values at boundary samples. Figs. 1a and 1b illustrate the issues of the IISPH boundary handling. Fig. 1c indicates the desired pressure distribution that we achieve with the proposed Pressure Boundaries.

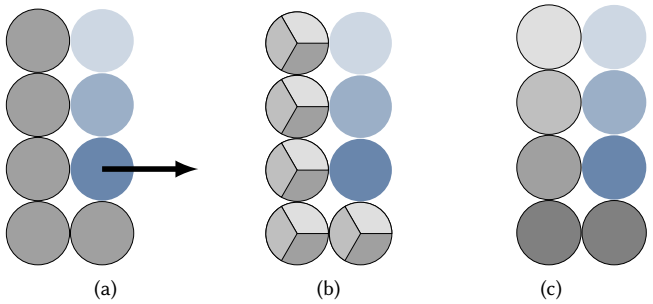


Fig. 1. The illustrations show boundary samples (gray) and fluid samples (blue). Pressure is color-coded. 1a: IISPH erroneously assumes equal pressure values at all boundary samples adjacent to the same fluid sample. The arrow indicates the limitation that the resulting pressure force exerted from boundary samples only varies in magnitude, but not in direction. 1b: At the same time, IISPH assumes several, potentially inconsistent pressure values at one boundary sample if this sample is adjacent to several fluid samples. 1c: In contrast, the proposed Pressure Boundary method computes unique and physically meaningful pressure values at boundary samples.

2 METHOD

Here, we first recapitulate the general idea of ISPH, followed by the specific PPE discretization of IISPH. Issues in the boundary handling of IISPH are explained and the proposed boundary handling is motivated. Afterwards, the concept and details of the proposed boundary handling are described.

2.1 ISPH

ISPH [Cummins and Rudman 1999] computes the pressure field p by solving a PPE of the form $\nabla^2 p = s$ with s being a source term that either encodes the divergence of a predicted velocity field, a predicted density deviation or a combination of both. As the pressure field is computed from a global formulation, it is typically rather smooth which positively affects the stability compared to, e.g., state-equation solvers [Adams et al. 2007; Monaghan 1994; Müller et al. 2005]. One option to derive a PPE with density invariance as source term is to start with the mass conservation law:

$$\frac{\mathrm{D}\rho(t+\Delta t)}{\mathrm{D}t} + \rho(t+\Delta t)\nabla \cdot \boldsymbol{v}(t+\Delta t) = 0\,. \tag{1}$$

$\boldsymbol{v}$ denotes the velocity, ρ denotes the density, t and Δt denote time and time step. Introducing the constraint $\rho(t+\Delta t) = \rho^0$, with ρ^0 denoting the rest density, using a backward difference for $\frac{\mathrm{D}\rho(t+\Delta t)}{\mathrm{D}t}$ and

$$\boldsymbol{v}(t+\Delta t) = \underbrace{\boldsymbol{v}(t) + \Delta t \boldsymbol{a}^{\text{non-p}}(t)}_{\boldsymbol{v}^*(t+\Delta t)} - \Delta t \frac{1}{\rho^0}\nabla p(t) \tag{2}$$

with $\boldsymbol{a}^{\text{non-p}}(t)$ denoting all non-pressure accelerations, we get

$$\frac{\rho(t+\Delta t) - \rho(t)}{\Delta t} + \rho^0\, \nabla \cdot \left(\boldsymbol{v}^*(t+\Delta t) - \Delta t \frac{1}{\rho^0}\nabla p(t)\right) = 0\,. \tag{3}$$

This equation can be written as

$$\frac{\rho^0 - \left(\rho(t) - \Delta t \rho^0 \nabla \cdot \boldsymbol{v}^*(t+\Delta t)\right)}{\Delta t} - \Delta t \nabla^2 p(t) = 0\,. \tag{4}$$

The term $\rho^*(t+\Delta t) = \rho(t) - \Delta t \rho^0 \nabla \cdot \boldsymbol{v}^*(t+\Delta t)$ is an approximation of the predicted density at time $t+\Delta t$, resulting in the following form of the PPE:

$$\Delta t^2 \nabla^2 p(t) = \rho^0 - \rho^*(t+\Delta t)\,. \tag{5}$$

Solving this PPE results in pressure values $p(t)$ and respective velocity changes $-\Delta t \frac{1}{\rho^0}\nabla p(t)$ that correct the predicted density deviation $\rho^0 - \rho^*(t+\Delta t)$ to make the fluid incompressible.

2.2 IISPH

IISPH [Ihmsen et al. 2014a] is a specific discretization of $\Delta t^2 \nabla^2 p(t)$ that is motivated by two reasons. First, it addresses the issue of operator inconsistency in SPH. As the SPH discretization of $\nabla^2 p$ is

generally not equal to the SPH discretization of $\nabla \cdot \nabla p$, i.e. $\langle \nabla^2 p \rangle \neq \langle \nabla \cdot \langle \nabla p \rangle \rangle$, IISPH discretizes $\langle \nabla \cdot \langle \nabla p \rangle \rangle$ rather than $\langle \nabla^2 p \rangle$.

According to [Cummins and Rudman 1999], the IISPH discretization is an accurate projection scheme which is in contrast to approximate projection schemes that discretize $\langle \nabla^2 p \rangle$. The convergence of IISPH and one particular approximate projection has been compared in, e.g. [Ihmsen et al. 2014a]. In the respective test scenarios, IISPH required less solver iterations than the approximate projection variant. Ihmsen et al. argue that the consideration of second-ring neighbors in IISPH improves its convergence. Another reason could be the fact that the same discretization $\langle \nabla p \rangle$ is consistently used in the computation of the pressure acceleration $\boldsymbol{a}^{\mathrm{p}}(t) = -\frac{1}{\rho^0} \nabla p(t)$. It can also be speculated that the formulation and the discretization of the source term in IISPH positively affect the iteration count. In addition to the comparatively low iteration count of the IISPH solver, a second motivation is the performance compared to other pressure solvers and the comparatively low memory consumption of IISPH in combination with a simple implementation. Although IISPH considers neighbors of neighbors, it can be implemented in a matrix-free way. The experiments in [Ihmsen et al. 2014a] indicate that IISPH is faster than iterative state-equation solvers for large scenarios, while the matrix-free implementation of the relaxed Jacobi technique for solving the PPE just requires the storage of seven additional scalar values per sample. Accordingly, IISPH uses the following two discretizations at a fluid sample position f:

$$-\frac{1}{\rho_f^0} \nabla p_f(t) = \boldsymbol{a}_f^{\mathrm{p}}(t) = -\sum_{f_f} m_{f_f} \left(\frac{p_f(t)}{\rho_f^2(t)} + \frac{p_{f_f}(t)}{\rho_{f_f}^2(t)} \right) \nabla W_{ff_f}(t) - \sum_{f_b} m_{f_b} \frac{p_f(t)}{\rho_f^2(t)} \nabla W_{ff_b}(t) \tag{6}$$

with f_f denoting fluid neighbors and f_b denoting boundary neighbors of f and

$$\Delta t^2 \nabla^2 p_f(t) = \Delta t^2 \sum_{f_f} m_{f_f} \left(\boldsymbol{a}_f^{\mathrm{p}}(t) - \boldsymbol{a}_{f_f}^{\mathrm{p}}(t) \right) \cdot \nabla W_{ff_f}(t) + \Delta t^2 \sum_{f_b} m_{f_b} \boldsymbol{a}_f^{\mathrm{p}}(t) \cdot \nabla W_{ff_b}(t) = \rho_f^0 - \rho_f^*(t + \Delta t) \tag{7}$$

which gives the PPE discretization.

Boundaries in IISPH are handled by pressure mirroring as proposed in, e.g., [Akinci et al. 2012]. As can be seen in Eq. (6), IISPH computes the pressure acceleration at fluid samples f due to boundary neighbors f_b with $\sum_{f_b} m_{f_b} \frac{p_f(t)}{\rho_f^2(t)} \nabla W_{ff_b}(t)$ which is an approximation of $\sum_{f_b} m_{f_b} \frac{p_{f_b}(t)}{\rho_{f_b}^2(t)} \nabla W_{ff_b}(t)$. This approximation is obtained from the mirroring assumptions $p_{f_b}(t) = p_f(t)$ and $\rho_{f_b}(t) = \rho_f(t)$. Further, since the computation of the pressure acceleration at a boundary sample is more involved (as all pressure forces at boundary samples have to be accumulated first to finally predict the next state of the boundary samples) IISPH uses the simplifying assumption that boundary samples b are not accelerated, i.e. $\boldsymbol{a}_b^{\mathrm{p}}(t) = \boldsymbol{0}$. This can be seen in Eq. (7) where the term $\Delta t^2 \sum_{f_b} m_{f_b} \boldsymbol{a}_f^{\mathrm{p}}(t) \cdot \nabla W_{ff_b}(t)$ is a simplification of $\Delta t^2 \sum_{f_b} m_{f_b} (\boldsymbol{a}_f^{\mathrm{p}}(t) - \boldsymbol{a}_{f_b}^{\mathrm{p}}(t)) \cdot \nabla W_{ff_b}(t)$ due to $\boldsymbol{a}_{f_b}^{\mathrm{p}}(t) = \boldsymbol{0}$. However, this simplification is correct for one-way coupled static and kinematic boundaries.

The mirroring assumptions lead to the artifacts illustrated in Fig. 1. From the perspective of a fluid sample, all adjacent boundary samples have the same pressure (see Fig. 1a). Further, different pressure values are considered at a boundary sample, if this sample has more than one fluid neighbor (see Fig. 1b).

2.3 IISPH with Pressure Boundaries

In order to avoid the illustrated artifacts and to realize a consistent pressure gradient at boundary samples (see Fig. 1c), we propose a novel PPE discretization that introduces changes to Eqs. (6) and (7). Sec. 2.3.1 derives the novel PPE discretization at fluid and boundary samples. Sec. 2.3.2 describes the computation, i.e. the implementation of all quantities that are required to solve the PPE. Sec. 2.3.3 motivates the employed notation where we prefer to work with sample volumes instead of densities.

2.3.1 Proposed PPE Discretization. In contrast to Eq. (6), we propose to compute the pressure acceleration at a fluid sample f in a unified way. Omitting time indices, we use

$$\boldsymbol{a}_f^{\mathrm{p}} = -\frac{V_f}{m_f} \sum_{f_j} V_{f_j} \left(p_f + p_{f_j} \right) \nabla W_{ff_j} \tag{8}$$

with V_f denoting the actual volume of sample f. This discretization has been used in, e.g. [Colagrossi and Landrini 2003] and is a variant of the so-called generalized SPH derivative operators as discussed, e.g. in [Price 2012]. An alternative derivation of this discretization is outlined in appendix A. Note that the sum in Eq. (8) processes all fluid and boundary neighbors f_j of the fluid sample f in the same way. Approximating assumptions on boundary pressures as in Eq. (6) are avoided and linear momentum is preserved.

In contrast to IISPH, we propose a PPE discretization that does not only consider unknown pressure at fluid samples, but also at boundary samples. I.e., the system consists of one equation per unknown pressure at a fluid sample and one equation per unknown pressure at a boundary sample. This novel formulation overcomes the mirroring assumptions of IISPH. It computes pressure values at boundary samples which are consistent with pressures at adjacent fluid samples (see Fig. 1c). At fluid samples, we use

$$\Delta t^2 \sum_{f_f} V_{f_f} \left(\boldsymbol{a}_f^{\mathrm{p}} - \boldsymbol{a}_{f_f}^{\mathrm{p}} \right) \cdot \nabla W_{ff_f} + \Delta t^2 \sum_{f_b} V_{f_b} \boldsymbol{a}_f^{\mathrm{p}} \cdot \nabla W_{ff_b} = 1 - \frac{V_f^0}{V_f} + \Delta t \nabla \cdot \boldsymbol{v}_f^* \,. \tag{9}$$

with V_f^0 denoting the rest volume of sample f. This equation is closely related to Eq. (7) and derived in appendix A. The velocity divergence $\nabla \cdot \boldsymbol{v}_f^*$ is discretized with Eq. (16). It is a variant of the generalized SPH derivative operators [Price 2012] and commonly used as SPH divergence operator. Despite the similarity to Eq. (7), Eqs. (7) and (9) differ in the computation of the pressure accelerations $\boldsymbol{a}_f^{\mathrm{p}}$ (Eq. (6) vs. Eq. (8)). The additional equations at boundary samples

can be derived from Eq. (9) by using $\boldsymbol{a}_b^{\mathrm{p}}(t) = \mathbf{0}$, as in IISPH:

$$-\Delta t^2 \sum_{b_f} V_{b_f} \boldsymbol{a}_{b_f}^{\mathrm{p}} \cdot \nabla W_{bb_f} = 1 - \frac{V_b^0}{V_b} + \Delta t \nabla \cdot \boldsymbol{v}_b^* . \tag{10}$$

Here, the velocity divergence $\nabla \cdot \boldsymbol{v}_b^*$ is discretized with Eq. (17) analogous to the divergence operator used in Eq. (9). Eqs. (9) and (10) constitute the proposed PPE discretization $\boldsymbol{Ap} = \boldsymbol{s}$ with $\boldsymbol{p}$ denoting the vector of all pressure values at fluid and boundary samples. Accordingly, Eqs. (9) and (10) can be denoted as $(\boldsymbol{Ap})_f = s_f$ and $(\boldsymbol{Ap})_b = s_b$, respectively.

2.3.2 Computation of Required Quantities. In order to solve the system, various quantities have to be computed: pressure accelerations $\boldsymbol{a}_f^{\mathrm{p}}$ at fluid samples, rest volumes V_f^0 of fluid and V_b^0 of boundary samples, actual volumes V_f of fluid and V_b of boundary samples and also velocity divergences $\nabla \cdot \boldsymbol{v}_f^*$ at fluid and $\nabla \cdot \boldsymbol{v}_b^*$ at boundary samples.

Rest volumes. The rest volume of a fluid sample is computed as

$$V_f^0 = h^3 \tag{11}$$

with h being the initial sample distance, i.e. the edge length of a cube-shaped fluid sample that is typically used for the initial sampling of the fluid body. The rest volume of a boundary sample is computed as

$$V_b^0 = \frac{\gamma}{\sum_{b_b} W_{bb_b}} . \tag{12}$$

It is intuitively clear that this computation only processes boundary neighbors as the rest volume of a boundary sample does not depend on possibly adjacent fluid samples. The coefficient γ is inspired by [Akinci et al. 2012] and accounts for an incomplete neighborhood of boundary samples as we only use one layer of boundary samples to represent the surface of boundaries. In our experiments, we use $\gamma = 0.7$ in combination with a cubic spline kernel [Monaghan 2005] with a smoothing length of $2h$. This is motivated by the fact that $\frac{0.7}{\sum_{b_b} W_{bb_b}} = h^3$ for boundary samples that are evenly sampled in a plane with distance h. Interestingly, Eq. (12) can consider adjacent boundary samples from different boundary objects. This allows the handling of intersecting objects. Eq. (12) can also be used to update the rest volume during a simulation if intersecting boundary objects move relative to each other.

Actual volumes. If all samples had the same rest volume, the actual volume of a sample could be computed as $V_i = \frac{1}{\sum_{i_j} W_{ii_j}}$. Due to the flexible boundary sampling, however, this is not the case. According to [Rosswog 2015], the volume of a sample can be computed as $V_i = \frac{V_i^0}{\sum_{i_j} V_{i_j}^0 W_{ii_j}}$ which results in

$$V_f = \frac{V_f^0}{\sum_{f_f} V_{f_f}^0 W_{ff_f} + \sum_{f_b} V_{f_b}^0 W_{ff_b}} \tag{13}$$

for the actual volume of a fluid sample. The actual volume of a boundary sample could be computed with the same equation. Analogous to the rest volume computation, however, we consider the incomplete neighborhood boundary samples and compute the actual volume as

$$V_b = \frac{V_b^0}{\sum_{b_f} V_{b_f}^0 W_{bb_f} + \sum_{b_b} V_{b_b}^0 W_{bb_b} + \beta} . \tag{14}$$

We use $\beta = 0.15 \cdot h^3$ which models a planar boundary sampling where only one side can be in contact with fluid. For planar boundaries where both sides can be in contact with fluid, the offset β would be zero. Assuming that the neighboring boundary samples b_b of boundary samples b are uniformly distributed on a plane, i.e. $V_b^0 = V_{b_b}^0$ and using Eq. (12) allows us to simplify Eq. (14) to

$$V_b = \frac{V_b^0}{\sum_{b_f} V_{b_f}^0 W_{bb_f} + \gamma + \beta} \tag{15}$$

This simplification saves memory and computation time as there is nothing to compute for boundary samples that have no fluid neighbors.

Velocity divergences. The velocity divergence at a fluid sample is computed as

$$\begin{aligned} \nabla \cdot \boldsymbol{v}_f^* = &- \sum_{f_f} V_{f_f} \left(\boldsymbol{v}_f^* - \boldsymbol{v}_{f_f}^* \right) \cdot \nabla W_{ff_f} \\ &- \sum_{f_b} V_{f_b} \left(\boldsymbol{v}_f^* - \boldsymbol{v}_{f_b}^* \right) \cdot \nabla W_{ff_b} . \end{aligned} \tag{16}$$

For one-way coupled static objects, the velocity $\boldsymbol{v}_{f_b}^*$ of a boundary neighbor f_b is zero. For one-way coupled kinematic objects, $\boldsymbol{v}_{f_b}^*$ is user-defined. Hence, one-way coupled objects are accurately handled. For two-way coupled dynamic objects without prescribed velocities, we make the simplifying assumption that the pressure solver does not change the velocity $\boldsymbol{v}_{f_b}^*$, i.e. $\boldsymbol{v}_{f_b}^*$ is computed by applying all non-pressure forces including collision handling to the respective solid object. This assumption introduces an error to the two-way coupling with solid objects. Nevertheless, it still allows for a plausible simulation of two-way coupled solids, e.g. see [Akinci et al. 2012; Bender and Koschier 2017; Ihmsen et al. 2014a]. The opposite of the pressure force at a fluid sample due to a boundary sample is simply applied to the boundary sample, accumulated and then applied to the solid object.

Eq. (16) considers the divergence with respect to adjacent fluid and boundary samples. In contrast, the computation of the velocity divergence at a boundary sample does not consider adjacent boundary samples. This corresponds to the fact that there is no relative movement between boundary samples. While this is true for samples of the same solid object, the assumption does not hold for adjacent boundary samples that belong to different objects. As the rest volume is updated with Eq. (12) in this case, however, there is generally no density change at a boundary sample due to the relative movement of boundary samples of another object. Therefore, it is correct to compute the velocity divergence at a boundary sample as

$$\nabla \cdot \boldsymbol{v}_b^* = - \sum_{b_f} V_{b_f} \left(\boldsymbol{v}_b^* - \boldsymbol{v}_{b_f}^* \right) \cdot \nabla W_{bb_f} . \tag{17}$$

Summary. Eqs. (11) to (17) describe the computations of all terms of the proposed PPE discretization $\boldsymbol{Ap} = \boldsymbol{s}$ in Eqs. (9) and (10).

2.3.3 Discussion of the Employed Notation. With our volume-centric notation, we follow, e.g., [Rosswog 2015; Solenthaler and Pajarola 2008]. As shown in [Solenthaler and Pajarola 2008], the density computation at interfaces requires special treatments which can be avoided by using the volume instead. Accordingly, [Solenthaler and Pajarola 2008] employs a volume-centric formulation for the pressure force.

2.4 Solver

Here, we follow [Ihmsen et al. 2014a] and use relaxed Jacobi. This is motivated by two reasons. On the one hand, [Ihmsen et al. 2014a; Takahashi et al. 2016] reported issues when using Conjugate Gradients. On the other hand, the usage of the same solver in IISPH and in the proposed pressure-boundary formulation enables comparisons that focus on the concept instead of the solver implementation. Pressure values p_i^0 are initialized with zero at all fluid and boundary samples and then iteratively updated with

$$p_i^{l+1} = \max\left(p_i^l + \omega_i \frac{s_i - (\boldsymbol{Ap}^l)_i}{a_{ii}}, 0\right) \tag{18}$$

where l denotes the iteration number. As negative pressures can cause instabilities due to attractive forces, we clamp negative pressure to zero in each iteration. The relaxation factor ω_i can vary for each sample. We use $\omega_i = 0.5\frac{V_i^0}{h^3}$ in our experiments, i.e. $\omega_f = 0.5$ and $\omega_b = 0.5\frac{\gamma}{h^3 \sum_{b_b} W_{bb_b}}$, which is motivated by the fact that we experienced convergence issues for constant relaxation factors in case of large volume ratios of fluid and boundary samples, i.e. very small boundary samples. The term a_{ii} indicates the i-th diagonal element of $\boldsymbol{A}$. Its derivation is outlined in appendix B. For a fluid sample, we get

$$a_{ff} = -\Delta t^2 \frac{V_f}{m_f} \left\| \sum_{f_j} V_{f_j} \nabla W_{ff_j} \right\|^2 - \Delta t^2 V_f \sum_{f_f} V_{f_f} \frac{V_{f_f}}{m_{f_f}} \left\| \nabla W_{ff_f} \right\|^2 \tag{19}$$

and for a boundary sample, we get

$$a_{bb} = -\Delta t^2 V_b \sum_{b_f} V_{b_f} \frac{V_{b_f}}{m_{b_f}} \left\| \nabla W_{bb_f} \right\|^2 . \tag{20}$$

Convergence criterion. In ISPH fluid simulations, the average deviation from the rest density is commonly used as the pressure solver's termination criterion, cf. [Ihmsen et al. 2014b]. However, our proposed form of the PPE also includes equations for boundary samples and therefore, boundary samples must be considered too in the evaluation of the solver's termination criterion. As mentioned above, we have a notion of a boundary sample's volume rather than its density. Thus, we terminate the Jacobi solver if it has reached a specified average volume error V^{error}. This error can be efficiently calculated in each Jacobi iteration from the residuals $r_i^l = \left(\boldsymbol{Ap}^l\right)_i - s_i$. The source term $s_i = 1 - \frac{V_i^0}{V_i} + \Delta t \nabla \cdot \boldsymbol{v}_i^*$ represents a relative volume deviation: the first part represents the current relative volume deviation, while the second term describes a predicted relative volume deviation due to the divergence of the predicted velocity $\boldsymbol{v}_i^*$. The solver computes a pressure field $\boldsymbol{p}^l$ such that $\boldsymbol{Ap}^l$ accounts for the relative volume deviation in the source term, i.e. $\left(\boldsymbol{Ap}^l\right)_i$ is also a relative volume deviation. Thus, we can compute the relative volume error of sample i from the residual r_i^l. Computing the average of all residuals of all samples provides us with the average relative volume error

$$V^{\text{error}} = \frac{1}{N} \sum_{i=1}^{N} \left| r_i^l \right| . \tag{21}$$

Please note that the volume error V^{error} can be easily converted to a density error and hence, both error variants are equivalent.

Algorithm 1 IISPH with Pressure Boundaries

procedure Compute Source Term
  for each boundary sample b **do**
    compute rest volume V_b^0 ▷ Eq. (12)
    compute volume V_b ▷ Eq. (15)
  for each fluid sample f **do**
    compute volume V_f ▷ Eq. (13)
  for each fluid sample f **do**
    predict velocity $\boldsymbol{v}_f^* = \boldsymbol{v}_f + \Delta t\, \boldsymbol{a}_f^{\text{non-p}}$
  for each sample i **do**
    compute source term s_i ▷ RHS of Eqs. (9) and (10)
    compute diagonal element a_{ii} ▷ Eqs. (19) and (20)
    initialize pressure p_i
procedure Solve PPE
  while not converged **do**
    for each fluid sample f **do**
      compute pressure acceleration $(\boldsymbol{a}_f^{\text{p}})^l$ ▷ Eq. (8)
    for each sample i **do**
      compute $(\boldsymbol{Ap}^l)_i$ ▷ LHS of Eqs. (9) and (10)
      update pressure p_i^{l+1} ▷ Eq. (18)
procedure Integration
  for each fluid sample f **do**
    $\boldsymbol{v}_f(t + \Delta t) = \boldsymbol{v}_f^* + \Delta t\, \boldsymbol{a}_f^{\text{p}}$
    $\boldsymbol{x}_f(t + \Delta t) = \boldsymbol{x}_f + \Delta t\, \boldsymbol{v}_f(t + \Delta t)$

2.5 Implementation

The term $(\boldsymbol{Ap}^l)_i$ is computed in two steps. First, the pressure accelerations $(\boldsymbol{a}_f^{\text{p}})^l$ are computed for fluid samples with Eq. (8). Then, the LHS of Eqs. (9) and (10) are computed to get $(\boldsymbol{Ap}^l)_f$ and $(\boldsymbol{Ap}^l)_b$ for fluid and boundary samples. Algorithm 1 summarizes the simulation update.

IISPH. To compare Pressure Boundaries with IISPH, we focus on the concept of computing individual pressure values for boundary samples rather than on specific implementation details. Therefore,

our implementation of IISPH is also based on the volume formulation of the PPE, i.e. we use Eq. (9) as the building block for the IISPH pressure solver instead of Eq. (7). It is also interesting to note that the pressure update in Eq. (18) is much easier to read and requires less floating point operations and thus is faster to compute than Eq. (16) in the original IISPH paper [Ihmsen et al. 2014a]. Therefore, we update the pressure values of fluid samples in our IISPH implementation as in Eq. (18). In summary, our implementation of IISPH is optimized in the same way as our implementation of Pressure Boundaries. This means that the presented performance gain factors in Sec. 3 would be larger if we would have used the original IISPH implementation from [Ihmsen et al. 2014a].

Differences to IISPH. In contrast to IISPH [Ihmsen et al. 2014a], we store only five additional scalar values per fluid sample instead of seven: one for the diagonal element a_{ff}, one for the source term s_f and three for the pressure acceleration $\boldsymbol{a}_f^{\mathrm{p}}$. Since we incorporate pressure values of boundary samples into the PPE, we require two additional loops over boundary samples in the ComputeSourceTerm procedure to compute their actual volumes, diagonal elements and source terms, i.e. we also have to store three additional scalar values for each boundary sample: the actual volume V_b, the diagonal element a_{bb} and the source term s_b. In the relaxed Jacobi iterations, we require one additional loop over boundary samples to update their pressure values. As the loops have no data dependencies, they are well suited for parallel architectures.

3 RESULTS

In this section, we compare the proposed Pressure Boundaries with standard IISPH [Ihmsen et al. 2014a]. All presented scenarios have been computed on a 12-core 2.6 GHz Intel Xeon E5-2690 with 32 GB of RAM.

In our implementation, we employ compact hashing [Ihmsen et al. 2011] for finding particle neighbors. For the SPH interpolation, we use the cubic spline kernel [Monaghan 2005] with a support of $2h$. Furthermore, we model surface tension and drag forces as proposed in [Akinci et al. 2013] and [Gissler et al. 2017], respectively. In order to demonstrate the applicability of our approach to two-way coupled dynamic objects, we integrated the Bullet physics library [Coumans, Erwin 2018] in our simulation framework. All computations are parallelized with Intel Threading Building Blocks [Pheatt 2008]. We used [FIFTY2 Technology 2018] to reconstruct the fluid surface. The ray-traced images were rendered with [Side Effects Software 2018].

3.1 Pillar

For the comparisons of the proposed Pressure Boundaries to standard IISPH, we use a fluid pillar within a box-shaped boundary of size $0.25\,\mathrm{m} \times 0.25\,\mathrm{m} \times 10\,\mathrm{m}$ and a sample size of $10^{-6}\,\mathrm{m}^3$. The scenario consists of 480 k fluid samples and 100 k boundary samples. The convergence criterion is set to a relative average volume error of 0.01 %. The rest density of the fluid is set to $1000\,\mathrm{kg\,m}^{-3}$. In all tests, we use XSPH viscosity [Schechter and Bridson 2012] with a coefficient of 0.05 and a time step of 0.1 ms, if not stated otherwise.

Pressure oscillations. Fig. 2 shows the average pressure over time for both methods. It can be seen that Pressure Boundaries reduce

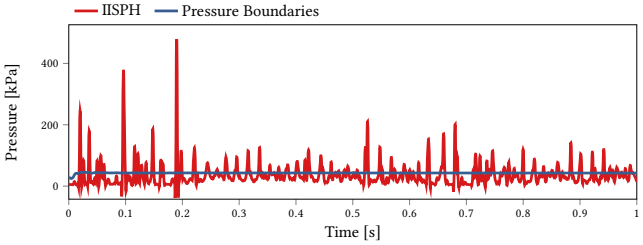


Fig. 2. Oscillations of the average pressure in the fluid pillar.

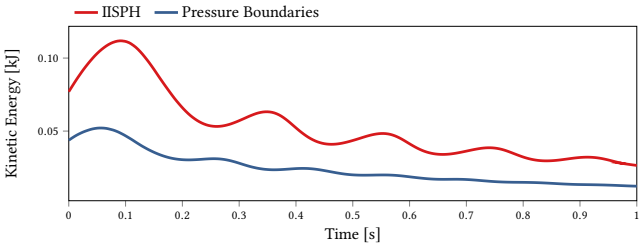


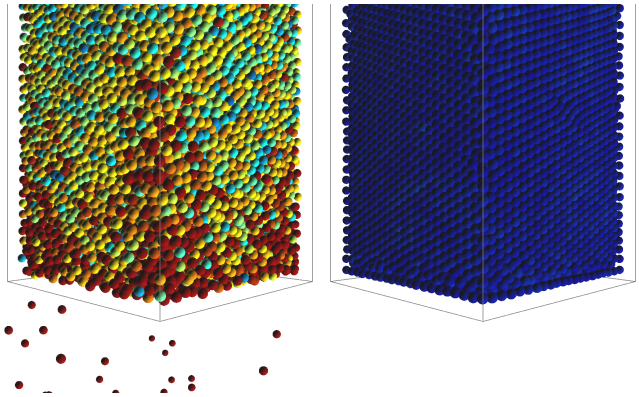
Fig. 3. Kinetic energy in the fluid pillar.

erroneous oscillations in the pressure field. In the same context, Fig. 3 shows improved values for the kinetic energy which is smaller and less fluctuating for Pressure Boundaries compared to IISPH.

Performance aspects. The reduced pressure oscillations result in an improved solver convergence for Pressure Boundaries. Tab. 1 shows that that IISPH requires about 37 iterations per simulation step, while the Pressure Boundary technique requires about 10 iterations. The reduced iteration count, however, is partially neutralized by the fact that Pressure Boundaries solve a larger PPE compared to IISPH. Nevertheless, the Pressure Boundary method computes the pressure field in 92 ms, while IISPH needs 320 ms. Fig. 4 indicates that Pressure Boundaries are stable for larger time steps compared to IISPH. In the pillar scenario, it is possible to increase the time step from 0.1 ms to 0.35 ms for Pressure Boundaries, while IISPH tends to get unstable for time steps larger than 0.1 ms.

Solver parameters. The relaxation coefficient ω governs the convergence of the Relaxed Jacobi solver in Eq. (18). As indicated in Fig. 5, larger values of ω reduce the required number of solver iterations. However, we could not arbitrarily increase ω, since only values in the range $0 < \omega \leq 0.5\frac{V_i^0}{h^3}$ resulted in stable solutions for our Pressure Boundary formulation. As shown in Fig. 6, the average and maximum pressure of the computed pressure field have approximately equal values for varying relaxation coefficients ω.

As shown in [Bender and Koschier 2017], the convergence of the solver can be improved by performing a warm start. For this purpose, a second solver parameter is used to initialize the pressure field. E.g., initial pressure values p_i^0 can be set as $p_i^0 = \lambda \cdot p_i(t - \Delta t)$ with $0 \leq \lambda \leq 1$. This parameter influences the solver's performance since larger values may reduce the iteration count. Note that $\lambda = 1$ works best for Pressure Boundaries, due to the more accurate pressure computation at boundary samples. In contrast to this, IISPH



(a) IISPH (b) Pressure Boundaries

Fig. 4. Pressure Boundaries work with a time step of up to 0.35 ms, while IISPH is unstable and suffers from leakage. Velocities are color-coded with blue corresponding to minimal and red corresponding to maximal velocity.

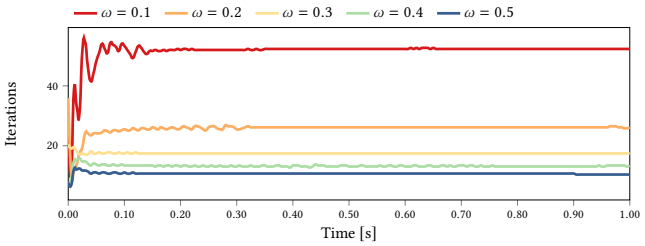


Fig. 5. Solver iterations for varying relaxation coefficients ω for the fluid pillar. We generally use a spatially adaptive relaxation coefficient $0 < \omega \leq 0.5\frac{V_i^0}{h^3}$. In the pillar scenario, however, all boundary and fluid samples have the same initial volume $V_i^0 = h^3$ which results in the same coefficient for all boundary and fluid samples.

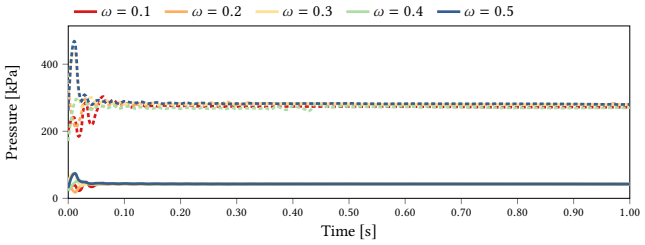


Fig. 6. Average and maximum (dashed) pressure for varying relaxation coefficients ω for the fluid pillar.

performs best when multiplying the pressure $p_i(t - \Delta t)$ of the last simulation step with $\lambda = 0.5$ [Bender and Koschier 2017; Ihmsen et al. 2014a].

3.2 Breaking Dam

In order to compare the required iteration counts of IISPH to our novel Pressure Boundaries approach, we simulated a breaking dam scenario inside a box-shaped domain of size 3 m×12 m×6 m with 6.4 million fluid and 625 k boundary samples. The particle spacing was 2 mm and we used different fixed time step sizes. Tab. 2 summarizes the iteration measurements for a simulation over ten seconds. In our experiment, we measured a smaller iteration ratio for smaller time step sizes, since often the minimum number of iteration was used. However, for larger time step sizes, due to more accurate pressure values at the boundary, Pressure Boundaries outperforms IISPH by a factor of about 5.58. This indicates that Pressure Boundaries scales better compared to IISPH for growing time steps. Furthermore, our approach computes a smooth pressure field over time, as indicated in Fig. 7.

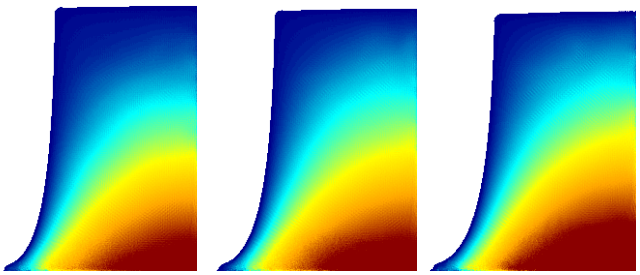


Fig. 7. Smooth color-coded pressure field for three consecutive frames of the Breaking Dam scenario simulated with our approach.

Two-way coupling. In another experiment, we placed several rigid bodies into the breaking dam scenario and simulated it for twenty seconds with a variable time step. The results are shown in Fig. 8 and in the accompanying video.

3.3 Tank

The pressure computation at boundary samples does not only positively affect the robustness and efficiency of the IISPH solver for animation purposes, but it can also be helpful in engineering applications. Fig. 9 illustrates a tank scenario (7 m × 1.05 m × 1.3 m) with different internal geometries with the aim to minimize sloshing. In contrast to standard IISPH, Pressure Boundaries enable the analysis of computed pressure values at the boundary. Each of the three scenarios consists of 17.5 million fluid samples and 1.3 million boundary samples. The particle spacing is 5 mm and the average time step is 0.47 ms. Each scenario is simulated for 15 seconds. The total computation time per simulation step is 5.1 s on average with Pressure Boundaries and 6.4 s with IISPH. Each PPE solve of our Pressure Boundaries approach requires 1.46 s with an average iteration count of 4.4, whereas IISPH requires 2.76 s and 10.2 iterations, respectively.

3.4 Gear

Pressure Boundaries are particularly appropriate for fast moving and complex boundary geometries. This is indicated in the scenario in Fig. 10 where two gears rotate with 1200 revolutions per minute. The scene is simulated for five seconds and consists of 5.5 million fluid samples and 750 k boundary samples. The particle spacing is 1.25 mm. Since the particles move very fast, considering the CFL condition, the size of the time step is small, i.e. 0.02 ms, resulting in

				iterations		avg. pressure computation time / Δt	
scene	particles	h	avg. Δt	IISPH	Pressure Boundaries	IISPH	Pressure Boundaries
Pillar	480 000	10 mm	0.10 ms	36.6	10.4	320 ms	92 ms
Breaking Dam	6.4 million	20 mm	1.50 ms	226.8	39.8	20 946 ms	3757 ms
Tank	17.5 million	5 mm	0.47 ms	10.2	4.4	2763 ms	1469 ms
Gear	5.5 million	1.25 mm	0.02 ms	2	2	264 ms	271 ms

Table 1. Measurements for the scenarios.

	iterations		avg. pressure computation time / Δt		ratio
Δt	IISPH	Pressure Boundaries	IISPH	Pressure Boundaries	IISPH / Pressure Boundaries
0.10 ms	2.0	2.0	185 ms	189 ms	0.98
0.15 ms	2.3	2.1	212 ms	198 ms	1.07
0.25 ms	5.6	3.4	518 ms	323 ms	1.60
0.50 ms	21.2	12.2	1958 ms	1152 ms	1.70
0.75 ms	44.2	24.3	4101 ms	2294 ms	1.78
1.00 ms	72.2	32.9	6668 ms	3106 ms	2.15
1.25 ms	104.5	36.7	9651 ms	3464 ms	2.79
1.50 ms	226.8	39.8	20 946 ms	3757 ms	**5.58**

Table 2. Comparison of IISPH with Pressure Boundaries using different time steps for the Breaking Dam scenario. The tolerated error was set to 0.01 %. The largest ratio in the pressure computation time is marked bold. For time steps larger than 1.5 ms, both IISPH and Pressure Boundaries suffer from leakage.

two iterations per simulation step. The total computation time per simulation step is 1.22 s on average and 271 ms for the SolvePPE procedure.

4 DISCUSSION

Performance. We have presented scenarios, where Pressure Boundaries works for larger time steps and requires less solver iterations compared to IISPH. As previously discussed in [Ihmsen et al. 2014b], however, it is not possible to draw a general conclusion about the performance differences of iterative pressure solvers. This is illustrated by the varying performance differences in Tab. 2 and by the range of performance differences in the selected scenarios summarized in Tab. 1.

Iteration count. The number of solver iterations scales with the time step size and pressure differences in the fluid. One extreme case would be a single fluid particle where standard state-equation SPH, e.g. [Becker and Teschner 2007; Monaghan 1992], certainly has the best performance. Even a huge number of fluid particles could be efficiently simulated with standard state-equation SPH as long as there is, e.g., just one layer of fluid particles on a planar boundary, i.e. small pressure differences. Another extreme case would be a scenario with much more boundary than fluid samples where the larger size of the Pressure Boundary PPE would increase the computation time per solver iteration compared to, e.g., standard IISPH.

Apart from the question whether iterative solvers are always the fastest ones, we would definitely always prefer such a solver over a non-iterative solver for many other reasons, e.g. simple parameterization, guaranteed density deviation, flexibility in terms of particle spacing and versatility in terms of scenarios. They are just easy-to-setup dependable workhorses.

Convergence. We can only speculate about aspects that positively influence the solver convergence. One aspect is perhaps the consideration of second-ring neighbors in the discretization. A second aspect could be the source term. It seems to be a difference whether current density deviations are taken into account or not. A velocity-divergence source term just considers the predicted velocity divergence, i.e. the predicted density deviation, while a density-invariant source term considers both, the predicted velocity divergence plus the current density error. It is also interesting to note that the pressure update in the first solver iteration with an initial pressure $\boldsymbol{p}^0 = 0$, i.e. $\lambda = 0$, is equivalent to a standard state equation. In this case, the divergence of the velocity change due to pressure accelerations would be $\boldsymbol{A}\boldsymbol{p}^0 = 0$ and the pressure in Eq. (18) would be updated with $p_i^1 = \max(\frac{\omega_i}{a_{ii}} s_i, 0)$. The term s_i is a relative volume error that can easily be translated to a density error and $\frac{\omega_i}{a_{ii}}$ corresponds to the stiffness constant of the state equation. This seems to indicate that even one solver iteration should give a reasonable result for a sufficiently small time step.

5 CONCLUSION AND FUTURE WORK

We have presented Pressure Boundaries for IISPH. In contrast to the previous IISPH boundary handling, approximate assumptions are avoided. Instead, boundary samples are incorporated into the PPE formulation which results in physically meaningful pressure values at boundary samples. In the description of Pressure Boundaries, we have used volume-centric SPH formulations that make the processing of arbitrarily sized boundary samples more intuitive compared

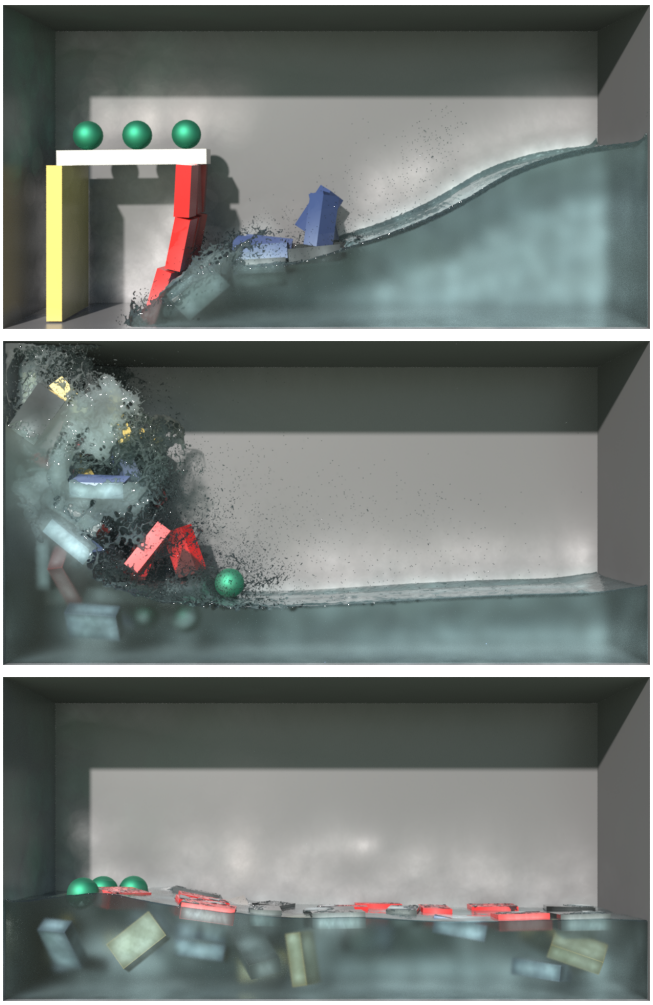


Fig. 8. Breaking Dam with 6.4 million fluid particles and two-way coupled solid objects simulated with our Pressure Boundaries approach. The average simulation time per time step was 1736 ms, whereof the pressure solver required 578 ms.

to density-centric formulations. The presented analyses show an improved solver convergence and larger possible time steps for Pressure Boundaries compared to the original IISPH boundary handling. Although Pressure Boundaries solves a larger PPE than the original IISPH, its computation is typically faster. While the description of Pressure Boundaries is based on IISPH, it would be interesting to investigate the utility of the proposed boundary handling in combination with other ISPH approaches that use alternative, but generally similar PPE formulations.

ACKNOWLEDGMENTS

This project is supported by the German Research Foundation (DFG) under contract number TE 632-1/2.

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. *ACM Transactions on Graphics* 26, 3 (2007).

Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2013. Versatile Surface Tension and Adhesion for SPH Fluids. *ACM Transactions on Graphics* 32, 6 (2013), 182:1–182:8.

Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. 2012. Versatile Rigid-fluid Coupling for Incompressible SPH. *ACM Transactions on Graphics* 31, 4 (2012), 62:1–62:8.

Stefan Band, Christoph Gissler, and Matthias Teschner. 2017. Moving Least Squares Boundaries for SPH Fluids. In *Virtual Reality Interactions and Physical Simulations*. Eurographics Association.

Markus Becker and Matthias Teschner. 2007. Weakly Compressible SPH for Free Surface Flows. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 209–217.

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.

Jan Bender and Dan Koschier. 2017. Divergence-Free SPH for Incompressible and Viscous Fluids. *IEEE Transactions on Visualization and Computer Graphics* 23, 3 (2017), 1193–1206.

Andrea Colagrossi and Maurizio Landrini. 2003. Numerical Simulation of Interfacial Flows by Smoothed Particle Hydrodynamics. *J. Comput. Phys.* 191, 2 (2003), 448–475.

Coumans, Erwin. 2018. The bullet physics library. www.bulletphysics.org. (2018).

Sharen J Cummins and Murray Rudman. 1999. An SPH Projection Method. *J. Comput. Phys.* 152, 2 (1999), 584–607.

FIFTY2 Technology. 2018. PreonLab. www.fifty2.eu. (2018).

Christoph Gissler, Stefan Band, Andreas Peer, Markus Ihmsen, and Matthias Teschner. 2017. Generalized drag force for particle-based simulations. *Computers & Graphics* 69 (2017). Issue 3.

Prashant Goswami, Philipp Schlegel, Barbara Solenthaler, and Renato Pajarola. 2010. Interactive SPH Simulation and Rendering on the GPU. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 55–64.

Xiaowei He, Ning Liu, Sheng Li, Hongan Wang, and Guoping Wang. 2012. Local Poisson SPH For Viscous Incompressible Fluids. *Computer Graphics Forum* 31, 6 (2012), 1948–1958.

Markus Ihmsen, Nadir Akinci, Markus Becker, and Matthias Teschner. 2011. A Parallel SPH Implementation on Multi-Core CPUs. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 99–112.

Markus Ihmsen, Jens Cornelis, Barbara Solenthaler, Christopher Horvath, and Matthias Teschner. 2014a. Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics* 20, 3 (2014), 426–435.

Markus Ihmsen, Jens Orthmann, Barbara Solenthaler, Andreas Kolb, and Matthias Teschner. 2014b. SPH Fluids in Computer Graphics. In *Eurographics (State of the Art Reports)*.

Dan Koschier and Jan Bender. 2017. Density Maps for Improved SPH Boundary Handling. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 1:1–1:10.

Shaofan Li and Wing Kam Liu. 2004. *Meshfree Particle Methods*. Springer-Verlag Berlin Heidelberg.

Joe J Monaghan. 1992. Smoothed Particle Hydrodynamics. *Annual Review of Astronomy and Astrophysics* 30, 1 (1992), 543–574.

Joe J Monaghan. 1994. Simulating Free Surface Flows with SPH. *J. Comput. Phys.* 110, 2 (1994), 399–406.

Joe 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. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 154–159.

Matthias Müller, Barbara Solenthaler, Richard Keiser, and Markus Gross. 2005. Particle-based Fluid-fluid Interaction. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 237–244.

Frank Ott and Erik Schnetter. 2003. A modified SPH approach for fluids with large density differences. In *ArXiv Physics e-prints*. 3112.

Chuck Pheatt. 2008. Intel® Threading Building Blocks. *Journal of Computing Sciences in Colleges* 23, 4 (2008), 298–298.

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

Stephan Rosswog. 2015. SPH Methods in the Modelling of Compact Objects. *Living Reviews in Computational Astrophysics* 1, Article 1 (2015). arXiv:astro-ph.IM/1406.4224

Hagit Schechter and Robert Bridson. 2012. Ghost SPH for Animating Water. *ACM Transactions on Graphics* 31, 4 (2012), 61:1–61:8.

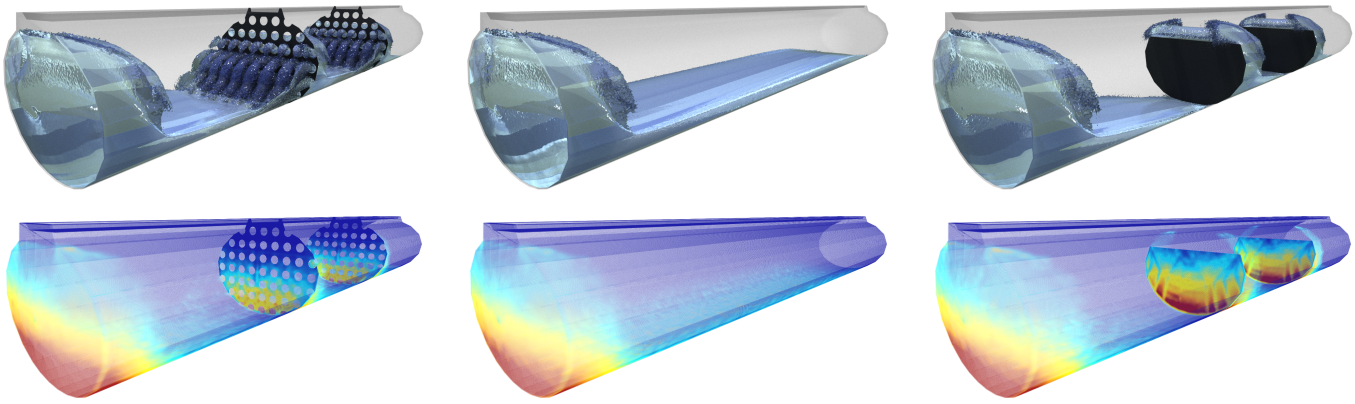


Fig. 9. Tank sloshing scenario with three different geometries (top row). Pressure Boundaries compute pressure values at solid boundaries (bottom row) with color-coded pressure (blue min, red max).

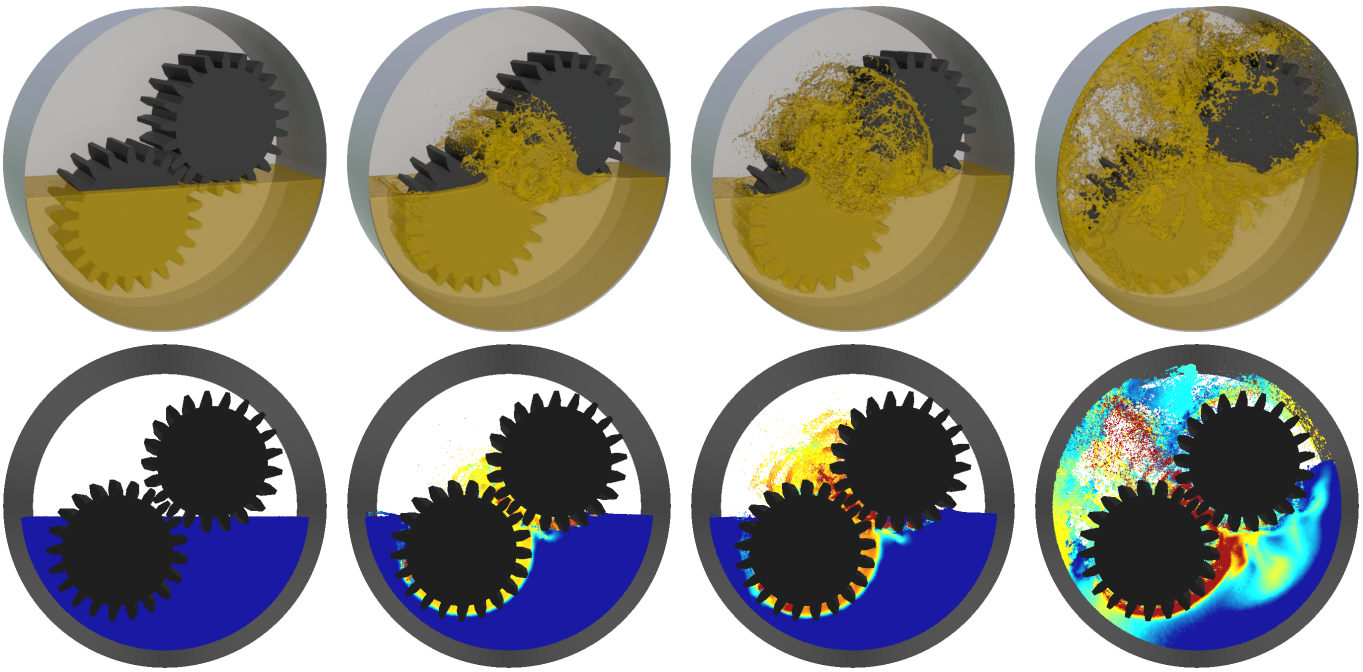


Fig. 10. Gear scene (top row) with color-coded velocities (bottom row). The Pressure Boundary approach is particularly appropriate for fast moving, geometrically complex boundaries.

Songdong Shao and Edmond Y.M. Lo. 2003. Incompressible SPH method for simulating Newtonian and non-Newtonian flows with a free surface. *Advances in Water Resources* 26, 7 (2003), 787 – 800.
Side Effects Software. 2018. Houdini. www.sidefx.com. (2018).
Barbara Solenthaler and Renato Pajarola. 2008. Density Contrast SPH Interfaces. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 211–218.
Barbara Solenthaler and Renato Pajarola. 2009. Predictive-corrective Incompressible SPH. *ACM Transactions on Graphics* 28, 3 (2009), 40:1–40:6.
Tetsuya Takahashi, Yoshinori Dobashi, Tomoyuki Nishita, and Ming C. Lin. 2016. An Efficient Hybrid Incompressible SPH Solver with Interface Handling for Boundary Conditions. *Computer Graphics Forum* (2016).
Mehmet Yildiz, R.A. Rook, and Afzal Suleman. 2009. SPH with the multiple boundary tangent method. *Internat. J. Numer. Methods Engrg.* 77, 10 (2009), 1416–1438.

A VOLUME-BASED FORMULATION

Pressure accelerations: Eq. (8) follows from

$$\nabla p_f = \nabla(1 \cdot p_f) = 1 \cdot \nabla p_f + p_f \cdot \nabla 1\,, \tag{22}$$

see, e.g., Eq. (2.31) in [Li and Liu 2004]. Applying the original SPH approximation results in

$$\begin{aligned}\nabla p_f &= \sum_{f_j} V_{f_j} p_{f_j} \nabla W_{ff_j} + p_f \sum_{f_j} V_{f_j} \nabla W_{ff_j} \\ &= \sum_{f_j} V_{f_j} \left(p_f + p_{f_j}\right) \nabla W_{ff_j} \,. \end{aligned} \tag{23}$$

PPE discretization: Eq. (9) can be derived, e.g., from the continuity equation at a fluid sample f:

$$\frac{\mathrm{D}\rho_f(t+\Delta t)}{\mathrm{D}t} + \rho_f^0 \nabla \cdot \boldsymbol{v}_f(t+\Delta t) = 0 \,. \tag{24}$$

Time discretization, assuming incompressibility $\rho_f(t+\Delta t) = \rho_f^0$, division by ρ_f^0 and using $\boldsymbol{v}_f(t+\Delta t) = \boldsymbol{v}_f^*(t+\Delta t) + \Delta t \boldsymbol{a}_f^{\mathrm{p}}(t)$ (see Eq. (2)), we get

$$\frac{1}{\Delta t} - \frac{\rho_f(t)}{\rho_f^0 \Delta t} + \nabla \cdot \boldsymbol{v}_f^*(t+\Delta t) + \nabla \cdot \Delta t \boldsymbol{a}_f^{\mathrm{p}}(t) = 0 \,. \tag{25}$$

Solving for $\nabla \cdot \Delta t \boldsymbol{a}_f^{\mathrm{p}}(t)$, discretizing it with SPH (see, e.g., Eq. (2.17) in [Monaghan 2005]) and omitting time indices, we get

$$\sum_{f_j} V_{f_j} \left(\Delta t \boldsymbol{a}_{f_f}^{\mathrm{p}} - \Delta t \boldsymbol{a}_f^{\mathrm{p}}\right) \cdot \nabla W_{ff_j} = -\frac{1}{\Delta t} + \frac{\rho_f}{\rho_f^0 \Delta t} - \nabla \cdot \boldsymbol{v}_f^* \,. \tag{26}$$

We multiply by $-\Delta t$, replace $\rho_f = \frac{m_f}{V_f}$ and $\rho_f^0 = \frac{m_f}{V_f^0}$ to finally get

$$\Delta t^2 \sum_{f_j} V_{f_j} \left(\boldsymbol{a}_f^{\mathrm{p}} - \boldsymbol{a}_{f_f}^{\mathrm{p}}\right) \cdot \nabla W_{ff_j} = 1 - \frac{V_f^0}{V_f} + \Delta t \nabla \cdot \boldsymbol{v}_f^* \,. \tag{27}$$

B DIAGONAL ELEMENTS

Fluid samples: We first extract the coefficients in the pressure accelerations in Eq. (8):

$$\begin{aligned}\boldsymbol{a}_f^{\mathrm{p}} =& -\frac{V_f}{m_f} \sum_{f_j} V_{f_j} \left(p_f + p_{f_j}\right) \nabla W_{ff_j} \\ =& -\frac{V_f}{m_f} \sum_{f_j} V_{f_j} p_f \nabla W_{ff_j} \\ & -\frac{V_f}{m_f} \sum_{f_j \neq f} V_{f_j} p_{f_j} \nabla W_{ff_j} - \frac{V_f}{m_f} V_f p_f \underbrace{\nabla W_{ff}}_{=0} \\ =& \underbrace{\left(-\frac{V_f}{m_f} \sum_{f_j} V_{f_j} \nabla W_{ff_j}\right)}_{\boldsymbol{c}_f} p_f \\ & + \sum_{f_j \neq f} \left[\underbrace{\left(-\frac{V_f}{m_f} V_{f_j} \nabla W_{ff_j}\right)}_{\boldsymbol{d}_{ff_j}} p_{f_j} \right] \\ =& \boldsymbol{c}_f p_f + \sum_{f_j \neq f} \left(\boldsymbol{d}_{ff_j} p_{f_j}\right) . \end{aligned} \tag{28}$$

This formulation is now applied to Eq. (9) and we get

$$\begin{aligned}(\boldsymbol{A}\boldsymbol{p})_f =& \Delta t^2 \sum_{f_f} V_{f_f} \left(\boldsymbol{a}_f^{\mathrm{p}} - \boldsymbol{a}_{f_f}^{\mathrm{p}}\right) \cdot \nabla W_{ff_f} \\ & + \Delta t^2 \sum_{f_b} V_{f_b} \boldsymbol{a}_f^{\mathrm{p}} \cdot \nabla W_{ff_b} \\ =& \Delta t^2 \sum_{f_f} V_{f_f} \left[\left(\boldsymbol{c}_f p_f + \sum_{f_j \neq f} \boldsymbol{d}_{ff_j} p_{f_j} \right) \right. \\ & \left. - \left(\boldsymbol{c}_{f_f} p_{f_f} + \sum_{f_{f_j} \neq f_f} \boldsymbol{d}_{f_f f_{f_j}} p_{f_{f_j}} \right) \right] \cdot \nabla W_{ff_f} \\ & + \Delta t^2 \sum_{f_b} V_{f_b} \left(\boldsymbol{c}_f p_f + \sum_{f_j \neq f} \boldsymbol{d}_{ff_j} p_{f_j} \right) \cdot \nabla W_{ff_b} \,. \end{aligned} \tag{29}$$

The pressure coefficient a_{ff}, i.e. the diagonal element of $\boldsymbol{A}$, follows as

$$\begin{aligned}a_{ff} =& \Delta t^2 \sum_{f_f} V_{f_f} \boldsymbol{c}_f \cdot \nabla W_{ff_f} \\ & - \Delta t^2 \sum_{f_f} V_{f_f} \boldsymbol{d}_{f_f f} \cdot \nabla W_{ff_f} \\ & + \Delta t^2 \sum_{f_b} V_{f_b} \boldsymbol{c}_f \cdot \nabla W_{ff_b} \,. \end{aligned} \tag{30}$$

The derivation of the coefficient from Eq. (29) is rather straightforward when we consider that there exist a j with $f_{f_j} = f$ and

that $\nabla W_{ff_f} = 0$ for $f_f = f$. The formulation in Eq. (30) requires the computation of a nested sum which can be avoided. Using the definitions of $\boldsymbol{c}$ and $\boldsymbol{d}$, we get

$$\begin{aligned} a_{ff} = &- \Delta t^2 \sum_{f_f} V_{f_f} \left(\frac{V_f}{m_f} \sum_{f_j} V_{f_j} \nabla W_{ff_j} \right) \cdot \nabla W_{ff_f} \\ &+ \Delta t^2 \sum_{f_f} V_{f_f} \frac{V_{f_f}}{m_{f_f}} V_f \nabla W_{f_f f} \cdot \nabla W_{ff_f} \\ &- \Delta t^2 \sum_{f_b} V_{f_b} \left(\frac{V_f}{m_f} \sum_{f_j} V_{f_j} \nabla W_{ff_j} \right) \cdot \nabla W_{ff_b} . \end{aligned} \tag{31}$$

The first sum of fluid neighbors and the third sum of boundary neighbors can be combined to a sum of all neighbors. Further, $\nabla W_{f_f f}$ is written as $-\nabla W_{ff_f}$:

$$\begin{aligned} a_{ff} = &- \Delta t^2 \frac{V_f}{m_f} \sum_{f_j} \left(\sum_{f_j} V_{f_j} \nabla W_{ff_j} \right) \cdot V_{f_j} \nabla W_{ff_j} \\ &- \Delta t^2 V_f \sum_{f_f} V_{f_f} \frac{V_{f_f}}{m_{f_f}} \nabla W_{ff_f} \cdot \nabla W_{ff_f} . \end{aligned} \tag{32}$$

We finally get the diagonal element of $\boldsymbol{A}$ for a fluid sample as

$$a_{ff} = -\Delta t^2 \frac{V_f}{m_f} \left\| \sum_{f_j} V_{f_j} \nabla W_{ff_j} \right\|^2 - \Delta t^2 V_f \sum_{f_f} V_{f_f} \frac{V_{f_f}}{m_{f_f}} \left\| \nabla W_{ff_f} \right\|^2 . \tag{33}$$

Boundary samples: We use the notation of Eq. (28) to rewrite Eq. (10) as

$$\begin{aligned} (\boldsymbol{Ap})_b = &- \Delta t^2 \sum_{b_f} V_{b_f} \boldsymbol{a}^{\mathrm{p}}_{b_f} \cdot \nabla W_{bb_f} \\ = &- \Delta t^2 \sum_{b_f} V_{b_f} \left(\boldsymbol{c}_{b_f} p_{b_f} + \sum_{b_{f_j} \neq b_f} \boldsymbol{d}_{b_f b_{f_j}} p_{b_{f_j}} \right) \cdot \nabla W_{bb_f} . \end{aligned} \tag{34}$$

As there exists a j with $b_{f_j} = b$, we get

$$a_{bb} = \Delta t^2 \sum_{b_f} V_{b_f} \boldsymbol{d}_{b_f b} \cdot \nabla W_{bb_f} = -\Delta t^2 V_b \sum_{b_f} V_{b_f} \frac{V_{b_f}}{m_{b_f}} \left\| \nabla W_{bb_f} \right\|^2 \tag{35}$$

for the diagonal element of $\boldsymbol{A}$ for a boundary sample.