Interlinked SPH Pressure Solvers for Strong Fluid-Rigid Coupling

CHRISTOPH GISSLER, University of Freiburg and FIFTY2 Technology GmbH
ANDREAS PEER and STEFAN BAND, University of Freiburg
JAN BENDER, RWTH Aachen University
MATTHIAS TESCHNER, University of Freiburg

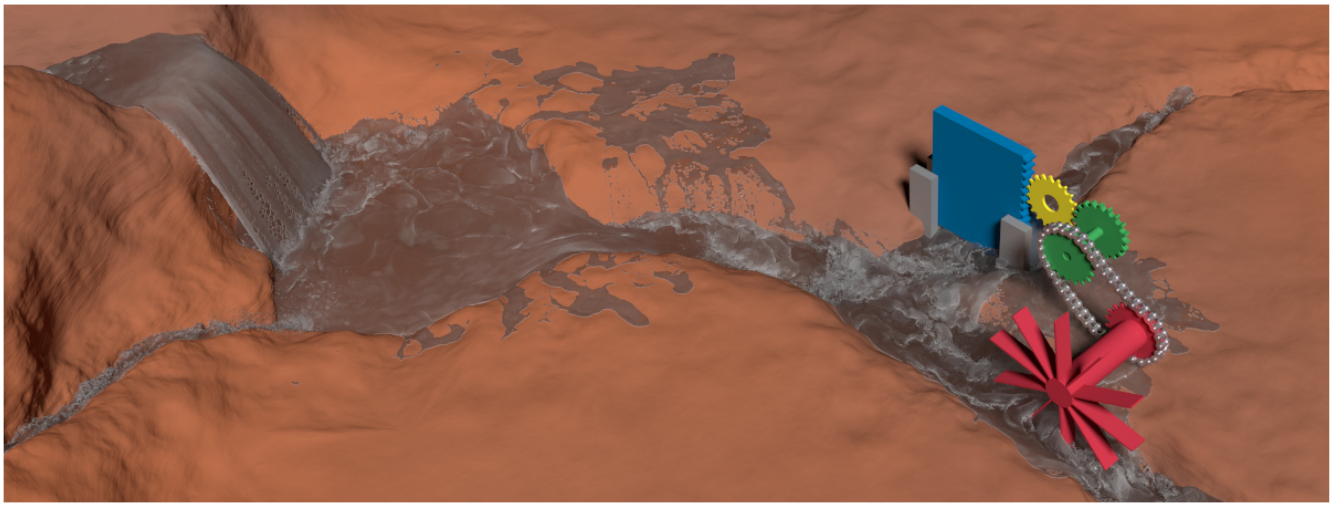

Fig. 1. SPH fluid with $43.8M$ particles in a terrain with $50M$ static rigid particles is two-way coupled with a water wheel that is connected to a gate via gears and a chain. Gears, chain, and water gate are represented with $2.3M$ dynamic rigid particles. Up to $90k$ simultaneous rigid-rigid contacts are handled.

We present a strong fluid-rigid coupling for Smoothed Particle Hydrodynamics (SPH) fluids and rigid bodies with particle-sampled surfaces. The approach interlinks the iterative pressure update at fluid particles with a second SPH solver that computes artificial pressure at rigid-body particles. The introduced SPH rigid-body solver models rigid-rigid contacts as artificial density deviations at rigid-body particles. The corresponding pressure is iteratively computed by solving a global formulation that is particularly useful for large numbers of rigid-rigid contacts. Compared to previous SPH coupling methods, the proposed concept stabilizes the fluid-rigid interface handling. It significantly reduces the computation times of SPH fluid simulations by enabling larger time steps. Performance gain factors of up to 58 compared to previous methods are presented. We illustrate the flexibility of the presented fluid-rigid coupling by integrating it into DFSPH, IISPH, and a recent SPH solver for highly viscous fluids. We further show its applicability to a recent SPH solver for elastic objects. Large scenarios with up to $90M$ particles of various interacting materials and complex contact geometries with up to $90k$ rigid-rigid contacts are shown. We demonstrate the competitiveness of our proposed rigid-body solver by comparing it to Bullet.

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

Additional Key Words and Phrases: Physically-based animation, fluid simulation, smoothed particle hydrodynamics, fluid-rigid coupling, rigid-rigid contacts

ACM Reference format:
Christoph Gissler, Andreas Peer, Stefan Band, Jan Bender, and Matthias Teschner. 2019. Interlinked SPH Pressure Solvers for Strong Fluid-Rigid Coupling. *ACM Trans. Graph.* 38, 1, Article 5 (January 2019), 13 pages.
https://doi.org/10.1145/3284980

Authors' addresses: C. Gissler, A. Peer, S. Band, and M. Teschner, Georges Köhler Allee 52, 79110 Freiburg im Breisgau, Germany; emails: {gisslerc, peera, bands, teschner}@informatik.uni-freiburg.de; J. Bender, RWTH Aachen, LuFG Computer Animation, 52056 Aache, Germany; email: bender@cs.rwth-aachen.de.

0730-0301/2019/01-ART5 $15.00
https://doi.org/10.1145/3284980

1 INTRODUCTION

Particle-sampled solids are a popular basis for the boundary handling in SPH fluid simulations (see, e.g., Ihmsen et al. (2014b)). The density computation at fluid particles considers contributions from nearby boundary particles and the pressure—derived from density deviations—induces interface forces. This concept works for one-way and two-way coupling, where the interface forces do not only affect the fluid velocity field but also the velocity of rigid bodies.

When implemented in iterative pressure solvers, the computed contact forces are often only applied after finishing the pressure

computation. For example, Akinci et al. (2012) use this concept in combination with PCISPH (Solenthaler and Pajarola 2009), Ihmsen et al. (2014a) in combination with IISPH, and Bender and Koschier (2017) in combination with DFSPH. In these works, the velocities of rigid particles are kept constant during the solver process that iteratively updates the fluid pressure. We show that this concept causes substantial issues in the two-way coupling for practically relevant scenarios. For example, instabilities can cause void regions at fluid-rigid interfaces and the simulation result is severely distorted. A simplistic solution would be a reduced time-step size. This, however, lowers the performance and is also difficult to implement as the reduced step size cannot be simply derived from the CFL condition.

Contribution: We propose to stabilize the handling of SPH fluid-rigid interfaces by using a strong fluid-rigid coupling, where the velocities of rigid-body particles are updated in each iteration of the fluid pressure solver. Our update considers fluid-rigid and rigid-rigid contacts that are uniformly handled with the SPH concept. While the fluid solver is responsible for the fluid-rigid contact forces, a novel system is solved to compute rigid-rigid contact forces. We therefore propose an approximate SPH formulation that works with particle-sampled rigid surfaces. If objects are in contact, then an artificial SPH density deviation is determined, which results in the respective pressure and contact forces that are computed using an implicit formulation.

We show that our strong coupling of fluid pressure and rigid-body velocities significantly reduces instabilities, i.e., void regions at fluid-solid interfaces. We further show scenarios where a similar simulation quality can only be achieved by previous boundary handling approaches, if the time-step size is reduced by one or two orders of magnitude, resulting in a performance gain factor of up to 58 for our method. Independently from the fluid-rigid coupling, we also demonstrate the competitiveness of our rigid-body solver by comparing it to Bullet (Coumans 2018) and showing stability advantages and performance gains by up to a factor of 9.

Our approach can be combined with iterative SPH pressure solvers and with a boundary handling that computes fluid-rigid interface forces. Although the proposed two-way coupling is tightly interconnected with the fluid solver, it can still flexibly be used with various SPH pressure solvers. We emphasize this by deriving our concept for a generic SPH fluid solver and by presenting experiments with four different solvers: IISPH, DFSPH, a solver for highly viscous SPH fluids (Peer et al. 2015), and a solver for elastic SPH solids (Peer et al. 2018). Our approach does not only handle large scenarios with up to $90M$ particles as shown in Figure 1It also handles complex contact geometries with up to $90k$ simultaneous rigid-rigid contacts.

2 RELATED WORK

The graphics community investigates the two-way coupling of fluids with solid objects in the context of various fluid-solver concepts, e.g., two-way coupling for height-field fluids (Chentanez and Müller 2010; Solenthaler et al. 2011; Thürey et al. 2007), for Lattice-Boltzmann fluids (Thürey et al. 2006), for the Lagrangian vortex method (Vines et al. 2014), for model-reduced fluids (Gerszewski et al. 2015), or for MPM fluids (Yan et al. 2018).

Traditionally, a lot of interesting approaches work with Eulerian fluids. For example, the fluid-rigid coupling proposed by Carlson et al. (2004), and also used by Kwatra et al. (2010) to simulate an articulated swimmer, treats regions occupied by rigid bodies as fluid during the pressure solve. Although solid regions are considered in the pressure solve, the rigidity constraint of these regions is only fulfilled in a second step. This is resolved by Klingner et al. (2006), where fluid and rigid constraints are simultaneously enforced. An extension to deformable solids is presented by Chentanez et al. (2006). Lentine et al. (2011) and Tan et al. (2011) both present an improved two-way coupling for articluated swimming creatures. A strong coupling with FLIP fluids is proposed by Batty et al. (2007), where rigid bodies are incorporated into the solver formulation. Rigid-rigid contacts, however, are handled sequentially in a separate step. The FLIP-based stream function solver presented by Ando et al. (2015) treats solid regions as fluid similar to Carlson et al. (2004), but presents a monolithic solver for the fluid-rigid coupling. Rigid-rigid contacts are not discussed. Guendelman et al. (2005) focus on the Eulerian coupling of deformable thin shells, where fluid and solids are solved separately. In contrast, Robinson-Mosher et al. (2008) introduce a monolithic solver based on an implicit formulation that also works for thin shells. In this context, Robinson-Mosher et al. (2009) propose an improved handling of tangential velocities at interfaces. Monolithic fluid-solid coupling and the investigation of the solvability of the respective systems are addressed, e.g., by Grétarsson et al. (2011), Patkar et al. (2016), and Robinson-Mosher et al. (2011). The method of Robinson-Mosher et al. (2011) is used and extended towards Chimera grids by English et al. (2013). Lu et al. (2016) also build on the approach of Robinson-Mosher et al. (2011) to couple fluids with reduced deformable objects. A recent cut-cell method for the strong coupling of Eulerian fluids and deformable bodies is presented by Zarifi and Batty (2017).

Our article focuses on Lagrangian fluid formulations. Müller et al. (2004) show the interaction of fluids and deformable objects. Fluid and rigid objects are solved sequentially by Clavet et al. (2005). In the rigid update step, the fluid particles are considered as rigid and their penetration into rigid bodies is resolved. Oger et al. (2006) integrate pressure values of the fluid particles over the surface of the rigid to compute the coupling forces. Keiser et al. (2006) treat rigid-body particles as fluid to compute interacting forces similar to the forces used by Carlson et al. (2004). As these approaches base on explicit formulations, the simulation time step is rather limited. A predictor-corrector scheme for two-way fluid-rigid coupling is proposed by Becker et al. (2009). Although the coupling bases on a global formulation to solve for valid relative velocities at interfaces, non-penetration is only enforced in a subsequent correction of particle positions. An alternative two-way coupling with pressure-based SPH boundary forces is introduced by Akinci et al. (2012). The force formulation is particularly useful for incomplete neighborhoods and non-uniform boundary samplings, i.e., one-layer boundaries with particles of varying size can be handled. Rigid-rigid contacts and predicted velocities of rigid objects are not updated during the fluid solve. Akinci et al. (2013b) extend the concept of Akinci et al. (2012) to elastic solids. Fujisawa and Miura (2015) propose to adapt the density computation of fluid particles such that no boundary particles are needed.

However, their approach does not support two-way coupling. Similarly, Koschier and Bender (2017) propose density maps instead of boundary particles to compute the influence of a rigid body onto the fluid. Here, two-way coupling is realized by mirroring the interface forces from the fluid to a rigid similar to the method of Akinci et al. (2012). Takahashi and Lin (2016) and Takahashi et al. (2017) classify particles depending on the boundary condition to improve the convergence of the fluid solver. The computed interface forces are similar to the ones of Akinci et al. (2012). Oh et al. (2009) propose an impulse-based fluid-rigid coupling, which also can be used to simulate solid-solid interactions with particles. The approach, however, is local and multiple simultaneous contacts are resolved subsequently ignoring previous results. Two-way coupling solutions with particle-sampled rigid objects are also used in frameworks for multiple materials, e.g. (Keiser et al. 2005; Macklin et al. 2014; Solenthaler et al. 2007). Recently, Akbay et al. (2018) proposed an improved two-way coupling, which is applicable to both Eulerian and Lagrangian fluids.

In contrast to the discussed coupling methods, our concept interconnects two SPH pressure solvers to simultaneously resolve fluid-rigid and rigid-rigid contacts. Rigid-rigid contacts are modeled as deviations of an artificial SPH density at rigid-body particles that represent the surface of a rigid body. A global formulation is solved to handle rigid-rigid contacts. The two-way coupling is tightly connected to the fluid solver, but can still be applied to various SPH pressure solvers.

3 METHOD

We first motivate the addressed two-way coupling issue in existing iterative SPH fluid solvers in Section 3.1. Section 3.2 introduces the concept to resolve this issue. In Section 3.3, we detail the fluid-rigid force formulations we use. The proposed rigid-body solver used for our two-way coupling is introduced in Section 3.4. Here, we first derive the implicit formulation that we solve. We then describe the SPH implementation of the respective solver. Finally, in Section 3.5, we explain the combination of our rigid-body solver with a generic iterative SPH pressure solver.

3.1 Generic Iterative Pressure Solver with the Boundary Handling of Akinci et al. (2012)

Iterative SPH pressure solvers such as PCISPH (Solenthaler and Pajarola 2009), LPSPH (He et al. 2012), IISPH (Ihmsen et al. 2014a), and DFSPH (Bender and Koschier 2017) compute a pressure field and velocity changes from pressure gradients to preserve the rest density of the fluid at all particles (see Algorithm 1). A predicted fluid velocity $\mathbf{v}_f^{*,l}$ is initialized by applying all non-pressure induced velocity changes and then refined in each solver iteration l. Velocities at rigid particles $\mathbf{v}_r^*$ are predicted accordingly, but not refined during the solver iterations according to the concept of Akinci et al. (2012).

The iterative refinement of fluid velocities $\mathbf{v}_f^{*,l}$ depends on a pressure-induced force $\mathbf{F}_f^{p,l}$, which in turn depends on the iteratively refined pressure field p_f^l. Finally, the refinement of the pressure field p_f^l depends on predicted fluid velocities $\mathbf{v}_f^{*,l}$ that are updated in each iteration and on predicted velocities of rigid particles $\mathbf{v}_r^*$ that are constant during the iterations.

Keeping rigid-body velocities constant during the iterations is a simplification that introduces errors in the pressure computation, i.e., in the computation of the final velocities of the fluid and the rigid objects. This is due to the fact that each iteration computes forces at the fluid-rigid interface, which are applied to fluid particles, but are neglected at rigid particles. The pressure-induced fluid-rigid interface forces at rigid particles $\mathbf{F}_r^{fr}$ are only applied after the iterations by using the final pressure field. Rigid-rigid contacts and the resulting forces $\mathbf{F}_r^{rr}$ are also computed and applied once after the solver iterations. This means that the predicted velocities at rigid particles $\mathbf{v}_r^*$ are erroneous during the solver iterations. As these velocities influence the pressure computation, the pressure field is negatively affected, which finally introduces errors to the velocity field of the fluid.

ALGORITHM 1: Generic iterative pressure solver with the boundary handling of Akinci et al. (2012). Predicted velocities of fluid particles $\mathbf{v}_f^{*,l}$ are updated in each iteration l, while rigid particle velocities $\mathbf{v}_r^*$ are kept constant. Please note that actual solvers might update more quantities or in a different order, which does not affect the concept.

1: Initialize $l = 0$, p_f^l, $\mathbf{v}_f^{*,l}$, $\mathbf{v}_r^*$
2: **while** Density deviation too large **do**
3: Compute fluid pressure forces $\mathbf{F}_f^{p,l+1}(p_f^l)$
4: Compute predicted fluid velocity $\mathbf{v}_f^{*,l+1}(\mathbf{F}_f^{p,l+1})$
5: Compute pressure $p_f^{l+1}(\mathbf{v}_f^{*,l+1}, \mathbf{v}_r^*)$
6: l +=1
7: Update fluid
8: Compute fluid-rigid interface forces $\mathbf{F}_r^{fr}(p_f^l)$
9: Compute rigid-rigid contact forces $\mathbf{F}_r^{rr}(\mathbf{v}_r^*)$
10: Update rigid bodies

3.2 Iterative Pressure Solver with Interleaved Fluid-Rigid Velocity Update

We propose an improved two-way coupling where forces at the fluid-rigid interface are not only applied to fluid particles but also to rigid particles in each iteration. That is, instead of working with a predicted velocity $\mathbf{v}_r^*$ at rigid particles that is constant during the iterations, we propose to update the velocity $\mathbf{v}_r^{*,l}$ in each iteration l. In contrast to Algorithm 1, interface forces $\mathbf{F}_r^{fr}$ and forces due to rigid-rigid contact $\mathbf{F}_r^{rr}$ refine velocities $\mathbf{v}_r^{*,l}$ of rigid particles per iteration as illustrated in Algorithm 2. This refinement now constitutes the actual effect of the interface forces not only onto the fluid velocities but also onto the velocities of the rigid bodies. The erroneous assumption of constant velocities of rigid particles as in Algorithm 1 is not applied. As the improved predicted velocities $\mathbf{v}_r^{*,l}$ influence the pressure refinement p_f^l, they also affect the fluid velocities $\mathbf{v}_f^{*,l}$.

3.3 Fluid-Rigid Interface Forces

The coupling between fluid and rigid objects requires the computation of the fluid-rigid interface force $\mathbf{F}_r^{fr}$ as detailed in the previous section. Multiple different options to compute such an interface force have been proposed in previous work, e.g., by Adami et al.

ALGORITHM 2: Generic iterative pressure solver with the proposed interleaved fluid-rigid velocity update. In contrast to Algorithm 1, predicted velocities of fluid and rigid particles—$\mathbf{v}_f^{*,l}$ and $\mathbf{v}_r^{*,l}$—are updated in each iteration l. As the updated predicted velocities of fluid and rigid particles influence the pressure refinement p_f^l, the interleaved velocity update results in an improved pressure field, which in turn results in an improved fluid velocity field and in an improved rigid-body movement compared to Algorithm 1.

```
Initialize l = 0, p_f^l, v_f^{*,l}, v_r^{*,l}
while Density deviation too large do
    Compute fluid pressure forces F_f^{p,l+1}(p_f^l)
    Compute predicted fluid velocity v_f^{*,l+1}(F_f^{p,l+1})
    Compute fluid-rigid interface forces F_r^{fr,l+1}(p_f^l)
    Compute rigid-rigid contact forces F_r^{rr,l+1}(v_r^{*,l})
    Compute predicted rigid velocity v_r^{*,l+1}(F_r^{fr,l+1}, F_r^{rr,l+1})
    Compute pressure p_f^{l+1}(v_f^{*,l+1}, v_r^{*,l+1})
    l +=1
Update fluid
Update rigid bodies
```

(2012), Akinci et al. (2012), Band et al. (2018a, 2018b, 2017), Koschier and Bender (2017), Monaghan (1994), and Schechter and Bridson (2012). Our strong coupling formulation does not rely on a specific fluid-rigid interface computation as long as a force can be computed per rigid particle. We integrated the forces proposed by Akinci et al. (2012) and Band et al. (2018b) and will shortly summarize them in the following.

Both interface forces determine a pressure value for each rigid particle in contact with at least one fluid particle. This pressure value is considered when computing a pressure gradient for the fluid particle using SPH. For two-way coupling, the pairwise force acting from a rigid particle onto a fluid particle is mirrored and assumed to act on the respective rigid. The approaches by Akinci et al. (2012) and Band et al. (2018b) differ in how they determine the pressure value of the rigid particle.

Akinci et al. (2012) determine the pressure value at a rigid particle by assuming that it has the same pressure value as the respective fluid particle. In contrast, Band et al. (2018b) determine the pressure value at a rigid particle by doing an extrapolation from the fluid pressure values onto the boundary. They employ MLS for this pressure extrapolation to improve the robustness to sampling disorder compared to an SPH interpolation.

3.4 Rigid Body Solver

Each iteration of the proposed solver requires the computation of fluid-rigid interface forces $\mathbf{F}_r^{\text{fr}}$ and forces due to rigid-rigid contact $\mathbf{F}_r^{\text{rr}}$. As discussed in the previous section, we employ the fluid-rigid interface forces proposed by Akinci et al. (2012) and Band et al. (2018b).

Motivated by a unified computation of all forces with SPH, we propose to compute the rigid-rigid contact forces $\mathbf{F}_r^{\text{rr}}$ based on artificial density deviations at rigid particles. We therefore assume an artificial rest density, e.g., $\rho_r^0 = 1$. If there is a rigid-rigid contact, then we calculate a density $\rho_r > \rho_r^0$. We then determine contact forces $\mathbf{F}_r^{\text{rr}}$ such that $\rho_r = \rho_r^0$ for all rigid particles after applying the forces $\mathbf{F}_r^{\text{rr}}$ to the rigid bodies and updating their state.

This is realized by computing an artificial pressure field p_r and deriving artificial pressure forces from the pressure gradient at all rigid particles. These forces cause velocity changes whose divergence cancels the density errors. In the following, we first derive the formulation to compute p_r in Section 3.4.1. Then, in Section 3.4.2, we show how to implement the presented solver with SPH.

3.4.1 System. The proposed system can be derived from the continuity equation $\frac{\mathrm{D}\rho_r}{\mathrm{D}t} = -\rho_r \nabla \cdot \mathbf{v}_r$ with ρ_r being the density and $\mathbf{v}_r$ being the velocity of a rigid particle r. Discretizing time with a backward difference and introducing the constraint that the density at time $t + \Delta t$ with Δt being the simulation time step should be equal to the desired rest density, i.e., $\rho_r^{\text{next}} = \rho_r^0$, we get

$$\frac{\rho_r^0 - \rho_r}{\Delta t} = -\rho_r \nabla \cdot \mathbf{v}_r^{\text{next}}, \tag{1}$$

with $\mathbf{v}_r^{\text{next}}$ being the desired velocity of a rigid particle at time $t + \Delta t$ to obtain the desired density ρ_r^0. Note that the velocity divergence at particle r is zero with respect to other particles of the same rigid body but may be non-zero with respect to particles of other rigid bodies. The velocity $\mathbf{v}_r^{\text{next}}$ can be written as

$$\mathbf{v}_r^{\text{next}} = \mathbf{v}_R^{\text{next}} + \boldsymbol{\omega}_R^{\text{next}} \times \mathbf{r}_r^{\text{next}}, \tag{2}$$

with $\mathbf{v}_R^{\text{next}}$ and $\boldsymbol{\omega}_R^{\text{next}}$ being the linear and angular velocities of the respective rigid body R at time $t + \Delta t$, respectively (see e.g., (Bender et al. 2014)). $\mathbf{r}_r^{\text{next}}$ denotes the vector from the center of mass of R to the position of the particle r at time $t + \Delta t$. Using Euler integration and Newton's second law, we write the linear velocity $\mathbf{v}_R^{\text{next}}$ as

$$\mathbf{v}_R^{\text{next}} = \mathbf{v}_R + \Delta \mathbf{t} \frac{\mathbf{1}}{M_R} \left(\mathbf{F}_R + \sum_k \mathbf{F}_k^{\text{rr}} \right), \tag{3}$$

with M_R being the mass of rigid body R. The force $\mathbf{F}_R$ comprises all momentum-changing sources except the unknown rigid-rigid contact forces $\mathbf{F}_r^{\text{rr}}$. This includes, e.g., gravitational force and the fluid-rigid interface forces. The sum considers all particles k of the rigid body R. The angular velocity $\boldsymbol{\omega}_R^{\text{next}}$ in Equation (2) can be written as

$$\boldsymbol{\omega}_R^{\text{next}} = \boldsymbol{\omega}_R + \Delta t \mathbf{I}_R^{-1} \left(\tau_R + (\mathbf{I}_R \boldsymbol{\omega}_R) \times \boldsymbol{\omega}_R + \sum_k \mathbf{r}_k \times \mathbf{F}_k^{\text{rr}} \right). \tag{4}$$

The torque τ_R contains all sources except the unknown rigid-rigid contact forces. Again, the fluid-rigid interface forces are included in τ_R. The matrix $\mathbf{I}_R$ is the inertia tensor of R and the sum considers all particles k of rigid body R. Using Equations (2) and (4), we write Equation (1) as

$$\frac{\rho_r^0 - \rho_r}{\Delta t} = -\rho_r \nabla \cdot \left(\mathbf{v}_R + \Delta t \frac{\mathbf{1}}{\mathbf{M}_R} \mathbf{F}_R \right) - \rho_r \nabla \cdot \left(\Delta t \frac{1}{M_R} \sum_k \mathbf{F}_k^{\mathrm{rr}} \right)$$
$$- \rho_r \nabla \cdot \left(\left(\boldsymbol{\omega}_R + \mathbf{I}_R^{-1} \left(\Delta t \boldsymbol{\tau}_R + \Delta t (\mathbf{I}_R \boldsymbol{\omega}_R) \times \boldsymbol{\omega}_R \right) \right) \times \mathbf{r}_r^{\mathrm{next}} \right)$$
$$- \rho_r \nabla \cdot \left(\Delta t \left(\mathbf{I}_R^{-1} \sum_k \mathbf{r}_k \times \mathbf{F}_k^{\mathrm{rr}} \right) \times \mathbf{r}_r^{\mathrm{next}} \right). \quad (5)$$

Compared to Equation (1), unknown velocities are replaced by unknown rigid-rigid contact forces in Equation (5). We introduce the approximation $\mathbf{r}_r^{\mathrm{next}} = \mathbf{r}_r$ and the term s_r with $s_r = \frac{\rho_r^0 - \rho_r}{\Delta t} + \rho_r \nabla \cdot \mathbf{v}_r^{\mathrm{s}}$ with $\mathbf{v}_r^{\mathrm{s}} = \mathbf{v}_R + \Delta t \frac{1}{M_R} \mathbf{F}_R + (\boldsymbol{\omega}_R + \mathbf{I}_R^{-1} (\Delta t \boldsymbol{\tau}_R + \Delta t (\mathbf{I}_R \boldsymbol{\omega}_R) \times \boldsymbol{\omega}_R)) \times \mathbf{r}_r$. Accordingly, s_r requires $\mathbf{F}_R$ and $\boldsymbol{\tau}_R$, which include the fluid-rigid interface forces and all forces we assume to be constant. They do not include the rigid-rigid contact forces. Finally, to get a system of equations, we move s_r to the left-hand side and leave the rigid-rigid contact force terms of Equation (1) on the right-hand side, which results in

$$s_r = -\rho_r \nabla \cdot \left(\Delta t \frac{1}{M_R} \sum_k \mathbf{F}_k^{\mathrm{rr}} + \left(\Delta t \mathbf{I}_R^{-1} \sum_k \mathbf{r}_k \times \mathbf{F}_k^{\mathrm{rr}} \right) \times \mathbf{r}_r \right), \quad (6)$$

which can be written as

$$s_r = -\rho_r \nabla \cdot \left(\Delta t \sum_k \mathbf{K}_{rk} \mathbf{F}_k^{\mathrm{rr}} \right), \quad (7)$$

with $\mathbf{K}_{rk} = \frac{1}{\mathbf{M}_R} \mathbb{1} - \tilde{\mathbf{r}}_r \mathbf{I}_R^{-1} \tilde{\mathbf{r}}_k$, where $\mathbb{1}$ is the identity matrix and $\tilde{\mathbf{r}}_r$ is the skew-symmetric cross-product matrix of vector $\mathbf{r}_r$. The matrix $\mathbf{K}$ is referred to as collision matrix and has been proposed by Mirtich (1996). Equation (7) applies to particle r of rigid body R and the sum considers all particles k of the same rigid body R. For two particles r and k, the term $\Delta t \mathbf{K}_{rk} \mathbf{F}_k^{\mathrm{rr}}$ gives the velocity change at particle r due to the contact force $\mathbf{F}_k^{\mathrm{rr}}$ at particle k. Equation (7) shares the same positive characteristics as other implicit formulations (e.g. (Xu et al. 2014)). In particular, large numbers of simultaneous contacts can be handled.

We model the rigid-rigid contact forces as pressure forces, i.e., we define $\mathbf{F}_k^{\mathrm{rr}} = -V_k \nabla p_k$, where V_k is an artificial volume of a rigid particle k and p_k is an unknown artificial pressure. Note that this artificial pressure value is only used for rigid-rigid contact resolution. It is independent of the pressure value at a rigid particle, which is determined for the computation of the fluid-rigid interface force. Now, Equation (7) can be written as

$$s_r = \rho_r \nabla \cdot \left(\Delta t \sum_k V_k \mathbf{K}_{rk} p_k \right), \quad (8)$$

with unknown artificial pressure. We have n equations with n unknown pressure values, where n is the overall number of all rigid particles of all rigid bodies. The equation for particle r considers pressure gradients ∇p_k for all particles k of the same object. If objects are in contact, then equation sets of different objects are coupled as ∇p_k is computed with particle pressures from the colliding objects. If an object is not in contact with any other object, then the respective equations can be removed from the system. In such a case, s_r is equal to zero for all particles of a body and $p_r = 0$ for those particles is a solution. The derived contact forces are also zero and none of the particles affects pressure gradients at other objects.

3.4.2 Implementation. This section describes the solver implementation for Equation (8). First, the discretization concept SPH is briefly outlined. Then, the SPH approximations for the quantities in Equation (8) are described. Finally, the relaxed Jacobi solver to compute the pressure is shown and its parameters are discussed.

SPH: We use SPH for interpolations $A_i = \sum_j \frac{m_j}{\rho_j} A_j W_{ij}$ where A is an arbitrary scalar quantity, m is the mass and ρ is the density of a particle (see, e.g. (Desbrun et al. 1996; Gingold and Monaghan 1977; Lucy 1977; Monaghan 2012; Müller et al. 2003; Stam and Fiume 1995)). W_{ij} is a kernel function, where we use the cubic spline kernel as, e.g., described by Monaghan (2005). The sum considers all particles j within a given distance to the position of sample i. In rest state, fluid particles are sampled at distance h and the support of our kernel function is $2h$. We also use SPH to compute spatial derivatives. In particular, we use $\nabla A_i = \rho_i \sum_j m_j (\frac{A_i}{\rho_i^2} + \frac{A_j}{\rho_j^2}) \nabla W_{ij}$ to approximate the gradient and $\nabla \cdot \mathbf{A}_i = \frac{1}{\rho_i} \sum_j m_j (\mathbf{A}_j - \mathbf{A}_i) \nabla W_{ij}$ to approximate the divergence. These approximations are commonly used. They are variants of the generalized SPH derivative operators as, e.g., discussed by Price (2012).

Artificial rest volume: Following, e.g., Solenthaler and Pajarola (2008) or Akinci et al. (2012), the rest volume V_r^0 of a rigid particle is computed as $V_r^0 = \frac{\gamma}{\sum_k W_{rk}}$. The sum considers rigid particles k of the same object within the kernel support at particle r. We use $\gamma = 0.7$. We motivate our choice of this value and demonstrate the result of changing γ in detail in Appendix A.

Artificial rest density: The artificial rest density ρ_r^0 of a rigid particle is set to one. As we parameterize the rigid-body solver with a relative density error at rigid particles, this choice is irrelevant for the simulation.

Artificial density: The actual density ρ_r of a rigid particle is computed as $\rho_r = \sum_k \rho_r^0 V_r^0 W_{rk}$. Here, the sum considers all particles k from all rigid bodies within the kernel support at r. If other objects are in contact at the location of particle r, then we get a density deviation, i.e., $\rho_r > \rho_r^0$.

Artificial volume: The actual volume V_r in case of $\rho_r > \rho_r^0$ is computed as $V_r = \frac{\rho_r^0 V_r^0}{\rho_r}$.

Computation of s_r: The computation of s_r requires the computation of $\mathbf{F}_R$ and $\boldsymbol{\tau}_R$, which include all forces and torques except the rigid-rigid contact forces. As our proposed rigid-body solver is integrated into an iterative fluid solver, s_r is updated in each iteration of the rigid solver due to the fact that the fluid-rigid interface forces are updated in each iteration of the fluid solver. All other forces contributing to $\mathbf{F}_R$ and $\boldsymbol{\tau}_R$ are assumed to be constant, and we compute them once in the initialization step of the rigid-body solver.

The term s_r further requires the divergence $\nabla \cdot \mathbf{v}_r^{\mathrm{s}}$, which is computed as

$$\nabla \cdot \mathbf{v}_r^{\mathrm{s}} = \frac{1}{\rho_r} \sum_k V_k \rho_k (\mathbf{v}_k^{\mathrm{s}} - \mathbf{v}_r^{\mathrm{s}}) \nabla W_{rk}. \quad (9)$$

The sum considers rigid neighbors k of r with particles k belonging to rigid bodies K, r to a rigid body R and $R \notin K$. This is due to the fact that the divergence term encodes density changes due to the relative movement of rigid bodies K in contact with R. For neighboring particles in R, the divergence term should be zero.

Computation of $\rho_r \nabla \cdot (\Delta t \sum_k V_k \mathbf{K}_{rk} \nabla p_k)$ of Equation (8): A first loop over all rigid particles computes $\nabla p_r = \rho_r \sum_k V_k \rho_k (\frac{p_r}{\rho_r^2} + \frac{p_k}{\rho_k^2}) \nabla W_{rk}$. The same loop accumulates $-\Delta t \frac{1}{M_R} V_r \nabla p_r$ and $-\Delta t \mathbf{I}_R^{-1} V_r \mathbf{r}_r \times \nabla p_r$ per rigid body and stores the result of the two sums as $\mathbf{v}_R^{\text{rr}}$ and $\boldsymbol{\omega}_R^{\text{rr}}$ at each rigid body, respectively. A second loop over all rigid particles computes the velocity $\mathbf{v}_r^{\text{rr}} = \mathbf{v}_R^{\text{rr}} + \boldsymbol{\omega}_R^{\text{rr}} \times \mathbf{r}_r$, i.e., $\mathbf{v}_r^{\text{rr}} = -\Delta t \sum_k V_k \mathbf{K}_{rk} \nabla p_k$. This is the velocity change at particle r due to the applied contact forces $\mathbf{F}_k^{\text{rr}}$ derived from ∇p_k at all particles k of the same rigid body. The second loop also computes the divergence $\rho_r \nabla \cdot (\Delta t \sum_k V_k \mathbf{K}_{rk} \nabla p_k) = -\rho_r \nabla \cdot \mathbf{v}_r^{\text{rr}}$ using $\rho_r \nabla \cdot \mathbf{v}_r^{\text{rr}} = \sum_k V_k \rho_k (\mathbf{v}_k^{\text{rr}} - \mathbf{v}_r^{\text{rr}}) \nabla W_{rk}$ with k being neighboring rigid particles of r.

Solver step: We use a relaxed Jacobi solver, i.e., we update pressure from iteration l to iteration $l+1$ using

$$p_r^{l+1} = p_r^l + \frac{\beta_r^{\text{RJ}}}{b_r} \left(s_r - \rho_r \nabla \cdot \left(\Delta t \sum_k V_k \mathbf{K}_{rk} \nabla p_k \right) \right), \qquad (10)$$

with β_r^{RJ} being the relaxation coefficient and b_r being the diagonal element of the linear system. Both variables are discussed below. Additionally, we clamp negative values of p_r^{l+1} to 0 to prevent attraction forces due to the SPH pressure gradient.

Computation of β_r^{RJ}: As basis for the relaxation coefficient β_r^{RJ}, we use a value of 0.5 as proposed by Ihmsen et al. (2014a). We additionally adapt its value to improve the convergence of our employed relaxed Jacobi solver for solving the rigid-rigid contacts. For this, we follow the idea of Tonge et al. (2012) and divide the relaxation factor by the overall number of contacts of the rigid body to which particle r belongs: $\beta_r^{\text{RJ}} = \frac{0.5}{num\_contacts}$.

Computation of b_r: We compute Equation (8) for all rigid-body particles in the scene and thus get a system of equations that can be written as $\mathbf{s} = \mathbf{B}\mathbf{p}$ with $\mathbf{s}$ and $\mathbf{p}$ denoting vectors that contain s_r and p_r of all rigid particles r, respectively. The relaxed Jacobi step in Equation (10) requires the diagonal elements of matrix $\mathbf{B}$ with b_r denoting the diagonal element in line r. Therefore, we accumulate all coefficients of p_r in Equation (8) to compute b_r.

Friction: We employ an explicit per-particle friction force based on the Coulomb model (see, e.g. (Bender et al. 2014)) to handle friction between rigid bodies. First, we compute a normal at each rigid particle r in contact as $\mathbf{n}_r = \mathbf{x}_r - \frac{\sum_k \mathbf{x}_k W_{rk}}{\sum_k W_{rk}}$. We furthermore compute the averaged relative predicted velocity due to other external forces at particle r and map this to a normal velocity and tangential velocity $\mathbf{v}_r^{\text{rel, t}}$ using $\mathbf{n}_r$. We estimate a contact impulse based on the normal velocity, which—together with the tangential velocity direction—allows us to compute a friction force using the Coulomb model. To ensure that the friction force only decreases the relative velocity of the whole rigid body, we clamp its value to a maximum of $-\frac{1}{\Delta t} \frac{1}{num\_contacts} \frac{1}{\mathbf{t}_r^T \mathbf{K}_{rr} \mathbf{t}_r} \mathbf{v}_r^{\text{rel, t}}$ per particle where $\mathbf{t}_r$ is the computed tangent at r.

3.5 Combining Fluid and Rigid Body Solver

The rigid-body solver computes the rigid-rigid contact forces and the predicted rigid velocities in lines 6 and 7 of the proposed concept in Algorithm 2. Algorithm 3 shows the same concept, extended by the embedding of the introduced rigid-body solver.

ALGORITHM 3: Our proposed two-way coupling. The introduced rigid-body solver is combined with a generic iterative SPH fluid solver using the concept shown in Algorithm 2. The computation of p_r^l is motivated in Section 3.4.1. Implementation details of s_r^l, $\mathbf{F}_r^{\text{rr},l}$, $\mathbf{v}_r^{*,l}$, and p_r^l are described in Section 3.4.2. The additional superscript l indicates that the quantities are updated in each iteration l of the solver.

1: Initialize $l = 0$, p_f^l, p_r^l, $\mathbf{v}_f^{*,l}$, $\mathbf{v}_r^{*,l}$
2: **while** Density deviation too large **do**
3: Compute fluid pressure forces $\mathbf{F}_f^{\text{p},l+1}(p_f^l)$
4: Compute predicted fluid velocity $\mathbf{v}_f^{*,l+1}(\mathbf{F}_f^{\text{p},l+1})$
5: Compute fluid-rigid interface forces $\mathbf{F}_r^{\text{fr},l+1}(p_f^l)$
6: Compute source term $s_r^{l+1}(\mathbf{F}_r^{\text{fr},l+1})$
7: Compute rigid-rigid contact forces $\mathbf{F}_r^{\text{rr},l+1} = -V_r \nabla p_r^l$
8: Compute predicted rigid velocity $\mathbf{v}_r^{*,l+1} = \mathbf{v}_r^{\text{s},l+1} + \mathbf{v}_r^{\text{rr},l+1}$
9: Compute pressure $p_f^{l+1}(\mathbf{v}_f^{*,l+1}, \mathbf{v}_r^{*,l+1})$
10: Compute pressure $p_r^{l+1}(s_r^{l+1}, p_r^l)$
11: $l += 1$
12: Update fluid
13: Update rigid bodies

The fluid and rigid-body solvers are closely coupled. The fluid solver provides fluid-rigid interface forces $\mathbf{F}_r^{\text{fr}}$ to the rigid-body solver. In particular, $\mathbf{F}_r^{\text{fr}}$ contributes to $\mathbf{F}_R$ and $\boldsymbol{\tau}_R$ used in $\mathbf{v}_r^{\text{s}}$ and finally in s_r (see Equation (9) and line 6 in Algorithm 3). In turn, the rigid-body solver computes updated predicted rigid velocities $\mathbf{v}_r^*$ that are used by the fluid solver. The fluid solver computes predicted density deviations from the divergence of predicted particle velocities, e.g., IISPH, or from predicted positions using predicted velocities, e.g., PCISPH. Thus, $\mathbf{v}_r^*$ influences the pressure computation at fluid particles.

The order of the rigid-body computations in lines 6, 7, 8, and 10 in Algorithm 3 is determined by the fact that these computations constitute a solver iteration for Equation (8). The term s_r in line 6 is the left-hand side of this equation, while $\mathbf{F}_r^{\text{rr}}$ in line 7 and $\mathbf{v}_r^*$ in line 8 are estimated during the computation of the right-hand side of Equation (8). Note that $\mathbf{v}_r^* = \mathbf{v}_r^{\text{s}} + \mathbf{v}_r^{\text{rr}}$ and $\mathbf{v}_r^{\text{rr}} = \Delta t \sum_k V_k \mathbf{K}_{rk} \nabla p_k$. After the computations in lines 6, 7 and 8, the solver step can be performed in line 10 using Equation (10).

The proposed concept interlinks two pressure solvers. As each update of the fluid solver affects the rigid solver and each rigid solver step influences the next fluid solver step, it is natural to couple them in a one-to-one manner as realized in Algorithm 3.

When solving both the fluid and the rigid-rigid collision in a coupled system as shown in Algorithm 3, a unified stopping criterion must be used. Our used stopping criterion is fulfilled when

the stopping criterion of our employed fluid solver is fulfilled and additionally the average density deviation of all rigid particles in contact with other rigid objects is smaller than 0.1%.

3.6 Discussion

We propose to couple our novel pressure-based rigid-body solver with an existing iterative fluid pressure solver to achieve strong coupling. By coupling them as shown in Algorithm 3, we no longer have a partitioned solver for fluid dynamics and rigid-body dynamics in contrast to the current state-of-the-art approach by Akinci et al. (2012). In fact, our approach describes a monolithic, unified system that solves fluid and rigid dynamics simultaneously. There is an equation for each fluid and rigid particle in the scene with an unknown pressure value for each respective particle.

Chentanez et al. (2006) and Klingner et al. (2006) initially proposed to solve fluid and solids simultaneously to achieve a strong coupling. In contrast to their approach, which uses an Eulerian fluid discretization and a Lagrangian solid discretization, we use a unified SPH-based discretization for fluids and rigid objects. For Position Based Dynamics methods, there exist different approaches to handle rigid-rigid and two-way coupling constraints. Macklin et al. (2014) first consider the rigid particles belonging to the same rigid object as being unconnected and then later constrain the movement of these particles to a rigid-body movement using shape matching. Instead, we directly consider the rigid-body movement in our system of equations. This is similar to the approach of Deul et al. (2014), who integrate the rigid-body motion into the rigid-rigid constraints.

4 RESULTS

We have combined our strong two-way coupling with two fluid solvers, IISPH (Ihmsen et al. 2014a) and DFSPH (Bender and Koschier 2017), with a solver for highly viscous fluids (Peer et al. 2015) and also with a solver for elastic solids (Peer et al. 2018). Vorticity is modeled with the micropolar formulation of Bender et al. (2017). Viscosity follows the idea of Morris et al. (1997). Fluid-air interaction in free-surface scenarios is modeled with the drag force proposed by Gissler et al. (2017). Surface tension is based on the method of Akinci et al. (2013a). Multi-phase simulations employ the number-density concept of Solenthaler and Pajarola (2008). We sample the surfaces of triangle meshes with boundary particles with an algorithm that is similar to the one proposed by Bell et al. (2005).

The experiments are organized as follows. Section 4.1 illustrates the improved stability and discusses the performance gain of the proposed fluid-rigid coupling compared to the state-of-the-art approach by Akinci et al. (2012). In Section 4.2, we demonstrate the utility of our approach in complex scenarios with large particle numbers, large numbers of interacting rigid bodies, large numbers of rigid-rigid contacts, and with multiple fluid phases. In Section 4.3, we compare our rigid-body solver with Bullet. Section 4.4 emphasizes the advantage of having a unified SPH-based rigid-body solver by showing phase transitions and by combining it with highly viscous fluids and elastic solids.

Table 1 summarizes particle numbers for all scenarios. Table 2 shows particle sizes, time steps, iteration counts, and computation time per frame. The experiments have been computed on a 16-core 3.1GHz Intel Xeon workstation.

Table 1. The Maximum Number of Particles and Maximum Simultaneous Rigid-Rigid Contacts Per Scene

		Particles			
			Solid		
Scene		Fluid	Static	Dynamic	Contacts
Rising sphere	Figure 2	4,886	304	43	0
Propeller pump	Figure 3	253k	86k	1,953	4
Orion splashdown	Figure 4	117k	29k	4,555	0
Moored buoys	Figure 5	5M	2M	149k	202
Armadillo drain	Figure 6	2M	407k	411k	6,325
AR scene	Figure 7	287k	850k	44k	210
Water gate	Figure 1	44M	50M	2.3M	90k
Nut and bolt	Figure 9	7.16M	85M	927k	4,007
Duck production	Figure 10	405k	477k	405k	374
Armadillo pool	Figure 11	1M	103k	58k	33
Valley	Figure 12	38M	19M	270k	3,739

For the armadillo pool and valley scene, the number of fluid particles also includes the particles of the viscous fluid and deformable objects.

Table 2. Particle Sizes, Maximum Time Steps, Average Solver Iterations, and Computation Time Per Frame for All Scenes

Scene	Part. size [mm]	Time step [ms]	Iterations	Time per frame
Rising sphere				
Akinci et al. (2012)	50	0.02	5	2.68s
Our concept	50	2	5	0.046s
Propeller pump				
Akinci et al. (2012)	50	0.3	2	6.63s
Our concept	50	3	14	1.43s
Orion splashdown				
Akinci et al. (2012)	200	2	3	1.31s
Our concept	200	20	19	0.39s
Moored buoys	20	0.1	25	18min
Armadillo drain	15	0.2	2	58s
AR scene				
Our concept	10	0.25	1	7.98s
Our concept	10	1	7	3.84s
Water gate	8	0.3	5	15min
Nut and bolt	10	0.1	2	19.9min
Duck production	5	0.1	1	13.2s
Armadillo pool	30	1	6	9s
Valley	50	1	8	6min

Comparative numbers for the method of Akinci et al. (2012) with similar simulation results are given for the first three scenes. For the armadillo pool and valley scene, pressure solver iterations are given, viscosity and elasticity solver iterations are not included.

4.1 Comparison to the Method of Akinci et al. (2012)

4.1.1 Rising Sphere. A 2D simulation of a sphere with a density of 100ρ rising in an IISPH fluid with a density of 1000ρ is used to illustrate stability improvements and performance gain of our concept compared to the method of Akinci et al. (2012). Figure 2(a) illustrates that their approach is unstable when using the same time

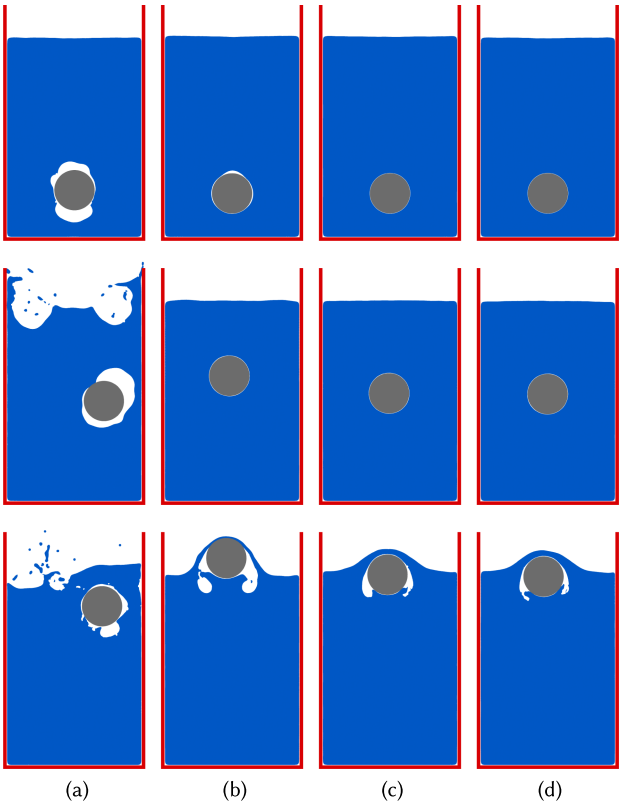

Fig. 2. 2D simulation of a rising sphere. (a), (b), and (c) show the method of Akinci et al. (2012) with time steps of 2ms, 0.2ms, and 0.02ms, respectively. (d) shows our approach with a time step of 2ms.

step as for our concept in Figure 2(d), which results in a stable simulation. Reducing the time step for the method of Akinci et al. (2012) in Figures 2(b) and 2(c) reduces artifacts and improves the stability, but is more expensive to compute. To get comparable simulation results to our approach, the time step has to be reduced by two orders of magnitude for the formulation of Akinci et al. (2012), resulting in a performance gain factor of 58 for our technique.

4.1.2 Propeller Pump. Figure 3 shows a scenario where a fast rotating propeller pumps fluid from one side of a tank to the other one. The propeller is two-way coupled, accelerated by a force and reaches a maximum speed of 160rpm. The DFSPH fluid has a density of 1000kgm^{-3}, the rubber ducks have a density of 500kgm^{-3} and the propeller has a density of 1500kgm^{-3}. The scene is simulated with a time step of 3ms using our strong coupling method. Using the weak coupling of Akinci et al. (2012) requires a reduced time step of 0.3ms to obtain a stable result. Accordingly, using our approach results in a performance gain factor of 4.6.

4.1.3 Orion Splashdown. The simulated splashdown of an Orion spacecraft capsule with the method of Akinci et al. (2012) and with our approach is compared in Figure 4. The density of the IISPH fluid is 1000ρ and the density of the capsule is 1200ρ. When using the same time step of 20ms for both approaches, our method is stable, while the weak coupling of Akinci et al. (2012) leads to an artificial void region below the capsule. This void region disturbs

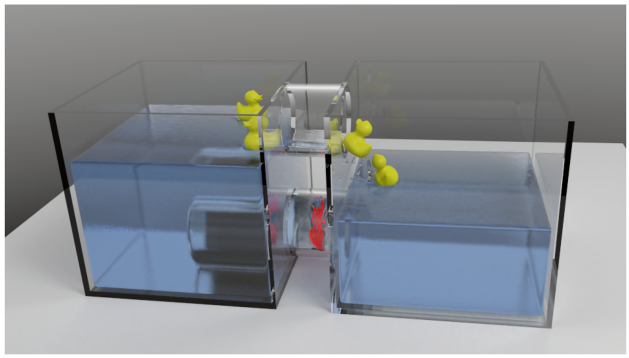

Fig. 3. A force-driven propeller pumps fluid from the right container to the left one using our approach. The time step is 3ms, while (Akinci et al. 2012) requires a time step of 0.3ms to obtain a stable simulation result.

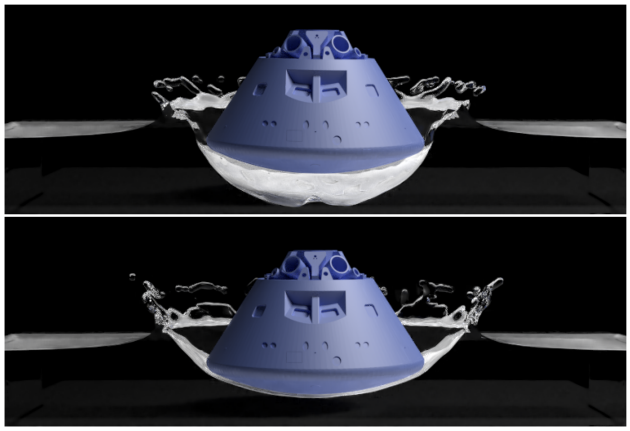

Fig. 4. Cutaway view of the Orion splashdown scene. The top image with a void region below the capsule is simulated using the method of Akinci et al. (2012). The bottom image without void region is simulated with our two-way coupling approach.

the interface handling and results in an unstable simulation. A stable result using their weak coupling can only be obtained with a reduced time step of 2ms, which results in a performance gain factor of 3.4 for our method.

4.2 Rigid Body Solver

4.2.1 Moored Buoys. The scene in Figure 5 illustrates our two-way coupling for multiphase fluids and also the rigid-rigid contact handling. Three buoys swim in two IISPH fluids with densities of 750ρ and 1500ρ. Since the three buoys have different densities (orange: 100ρ, green: 1100ρ, turquois: 2000ρ), they swim at different levels. The surfaces of the buoys and of the chain elements are particle-sampled. All rigid-rigid contacts of the respective chain elements and the buoys are handled by our proposed rigid-body solver.

4.2.2 Armadillo Drain. Figure 6 illustrates that our SPH-based rigid-body solver can simulate a large number of interacting, geometrically complex rigid bodies. Additionally, the 1500 Armadillos in this scene are two-way coupled with a DFSPH fluid. The particle

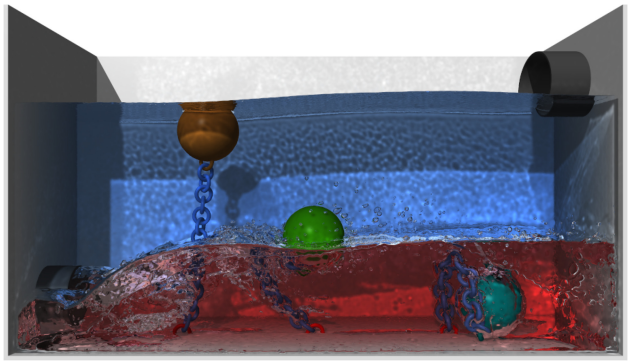

Fig. 5. Three buoys with different densities moored to the ground of a tank and interacting with two fluids of different densities.

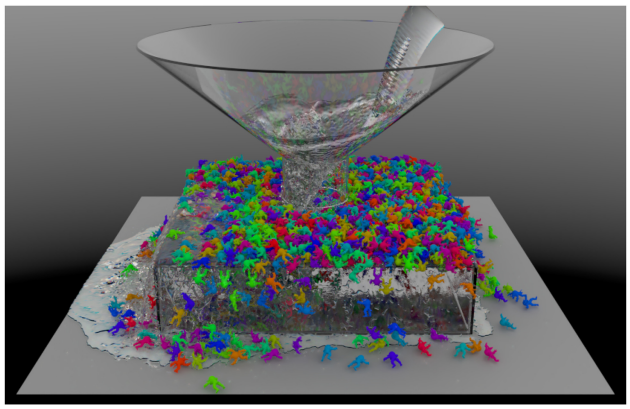

Fig. 6. 1,500 Armadillos are dropped into a funnel, which is then rinsed with water.

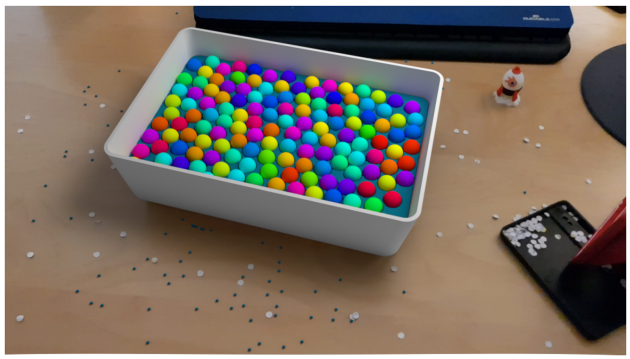

Fig. 7. Spheres are dropped into a pool of fluid. In this scene, we use the fluid-rigid interface force proposed by Band et al. (2018b).

numbers for the fluid and the Armadillos and also the number of simultaneous contact points are given in Table 1.

4.2.3 AR Scene. In the scene shown in Figure 7, we use the fluid-rigid interface force proposed by Band et al. (2018b) in contrast to the interface force by Akinci et al. (2012), which we use in other scenes. This demonstrates that we can use our strong coupling approach with different fluid-rigid interface forces.

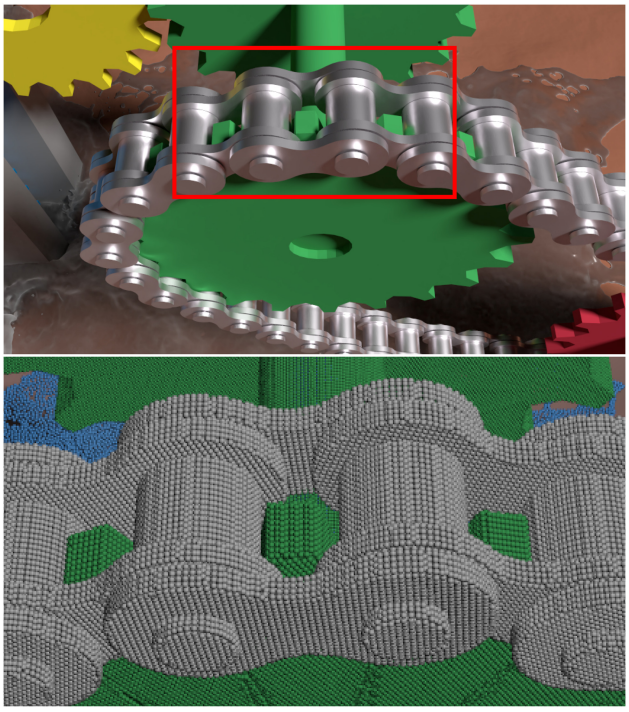

Fig. 8. These images show the chain in the water gate scene (see Figure 1). The bottom image shows a closeup, visualizing the rigid-body particles of the chain elements and the gear.

Furthermore, we simulated the scene with different time steps as shown in Table 2 to demonstrate that it is generally beneficial for the overall computation time to use larger time steps.

4.2.4 Water Gate. Figure 1 illustrates that our method can handle large-scale scenarios with millions of fluid and rigid-body particles and also with tens of thousands of simultaneous rigid-rigid contacts. A water wheel is connected to a water gate with a chain and multiple gears. The chain and the gears are fully simulated with our rigid-body solver. The connections between the respective chain elements are purely solved based on rigid-rigid contacts. All dynamic rigid bodies are two-way coupled with an IISPH fluid. Figure 8 shows a close-up of the chain and its particle representation.

4.2.5 Nut and Bolt. Our rigid-body solver stably simulates complex, interacting, concave geometry with many simultaneous contacts, which is demonstrated in the scene shown in Figure 9. We simulated a bolt inside a nut acting as a valve for water. The bolt is fully modeled as rigid body with six degrees of freedom. Using a screwdriver, the bolt is slowly loosened until it falls out. In this scene, we employed the fluid-rigid interface force proposed by Band et al. (2018b).

4.3 Comparison to Bullet (Coumans 2018)

To evaluate our rigid-body solver independently of the proposed strong fluid-rigid coupling, we compare it to Bullet (Coumans 2018), which is a popular state-of-the-art rigid-body engine that implements a PGS LCP solver (Bender et al. 2014).

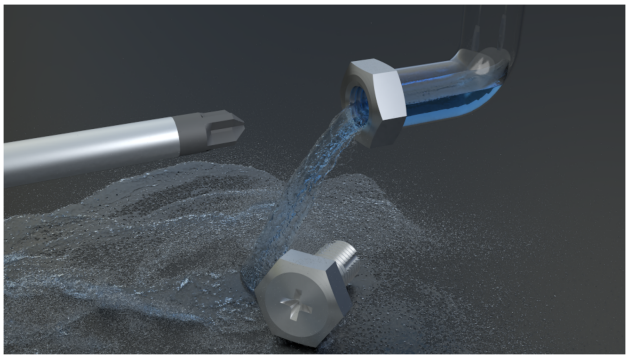

Fig. 9. A nut that acts as a valve for water is slowly loosened by a screwdriver until it falls out. The bolt is simulated as rigid body with six degrees of freedom using our approach.

Table 3. Timings for the Armadillo Drain Comparison Scene

Scene	Discretization per Armadillo	Time per frame
Armadillo drain		
Bullet	1 concave hull	0.23min
Bullet	36 concave hulls	1.92min
Ours	274 particles	1.44min

Table 4. Timings for the Nut and Bolt Comparison Scene

Scene	Time step [ms]	Stability	Time per frame
Nut and bolt			
Bullet	1	unstable	0.4s
Bullet	0.04	unstable	2.6s
Bullet	0.03	stable	3.8s
Ours	10	stable	0.4s

4.3.1 Armadillo Drain. We adapted the Armadillo drain scene shown in Section 4.2.2 by removing the fluid to compare our rigid-body solver with Bullet (Coumans 2018). The resulting timings from the comparison are shown in Table 3. Bullet is faster than our solver when discretizing the Armadillo with a single convex hull. However, this simplifies the problem drastically, since the Armadillo mesh is highly concave. When using 36 convex hulls, Bullet's overall computation time is slower compared to our solver, where we discretized the Armadillo mesh with 274 boundary particles.

4.3.2 Nut and Bolt. We use the nut and bolt from the scene shown in Figure 9 for a comparison with interacting geometry with a lot of simultaneous contact points. We only simulated the bolt being affected by gravity and resting inside the nut. This is a similar setup to the scene used by Xu et al. (2014) to compare their proposed rigid-body solver to Bullet. As proposed by them, we also employ a convex decomposition of the meshes when simulating them with Bullet.

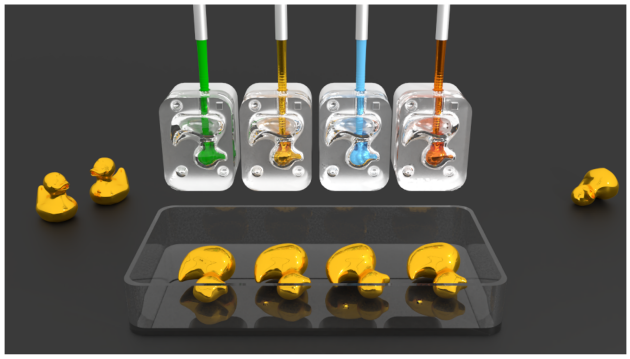

Fig. 10. Duck production scene. First, fluid is filled into the molds. The fluid particles are then transformed into a rigid body and simulated with our proposed rigid-body solver. Finally, the particles are again transformed into fluid.

Table 4 shows the result of the comparison. Using our proposed rigid-body solver, we were able to stably simulate the 4s of physical time in 1min and 21s. For the same overall computation time, Bullet does not produce a stable simulation as shown in the accompanying video. For Bullet to stably simulate this scene, we needed to reduce the time step to 0.03ms, which resulted in a computation time, which was larger by a factor of 9.5 compared to using our solver.

In addition to the computation time disadvantage of Bullet, we needed to create the convex decomposition of the nut and bolt meshes. We were not able to get an automatic convex decomposition of the meshes, which was accurate enough for the use with Bullet. Instead, we needed to manually separate the meshes into convex parts, which was time consuming. In contrast, with our rigid-body solver, only the surface of the rigid body needs to be sampled with particles, which can be robustly done, e.g., with the algorithm proposed by Bell et al. (2005).

4.3.3 Discussion. We have shown two examples in which our proposed rigid-body solver is faster and more stable than Bullet, which shows that our proposed rigid-body solver is competitive. However, obviously, one could construct scenes where our solver has disadvantages compared to Bullet. For example, our solver may use a lot of particles to discretize a simple box depending on the particle resolution. Accordingly, Bullet could be faster in scenes where only simple rigid-body objects like boxes are involved for which Bullet uses specialized collision algorithms. In contrast, our solver is especially well suited for many interacting, complex, and concave rigid bodies.

4.4 Unified SPH Solver

4.4.1 Duck Production. The duck production scenario in Figure 10 shows a setup where fluids solidify to rigid bodies and later become fluid again. This shows an advantage of having a unified solver based on SPH for fluid dynamics and rigid-body dynamics. It allows to switch the existing fluid particles to be simulated as a rigid body and back to be simulated as fluid without instabilities due to the conversion. Furthermore, since everything is simulated based on particles, it is not necessary to have a triangle mesh or to

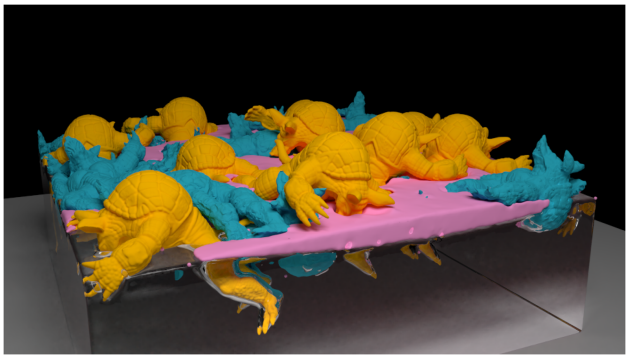

Fig. 11. Highly viscous (pink), elastic (turquoise), and rigid (orange) Armadillos are dropped into a pool of water.

Fig. 12. Creatures break through a wall while fleeing from a flood. Please also refer to the accompanying video to see the highly viscous mud and the elastic tree in this scene.

be able to create a collision shape, which would be needed when simulating this scene with for example Bullet (Coumans 2018).

4.4.2 Armadillo Pool. To illustrate the flexibility of our concept, various highly viscous (Peer et al. 2015), elastic (Peer et al. 2018), and rigid Armadillos (simulated with our rigid-body solver) are two-way coupled with an IISPH fluid. The fluid-viscous and fluid-elastic coupling of the highly viscous and elastic Armadillos is realized using the methods of Peer et al. (2015) and Peer et al. (2018), respectively. Our approach is used for the coupling of the rigid Armadillos to the fluid phase and to the viscous and elastic Armadillos. An image of the scene is shown in Figure 11.

4.4.3 Valley. Figure 12 shows the interactions of different materials in a large-scale scenario. Two animated creatures flee from an IISPH fluid. They need to run through highly viscous mud simulated using the approach of Peer et al. (2015) and break through a wall. The two-way coupling of the IISPH fluid with the bricks, the two-way coupling of the bricks with the highly viscous mud, the two-way coupling of bricks with the elastic tree (Peer et al. 2018), and the rigid-rigid contacts between the bricks are handled by our approach.

5 LIMITATIONS

The accuracy of the rigid-rigid contact handling is related to the size of a rigid-body particle, which in turn is related to the size of a fluid particle. Thus, the accuracy of the contact handling is coupled to the accuracy of the fluid simulation. Our approach can handle rigid-body particle sizes that vary to some extent, but we assume that they are generally similar to the size of a fluid particle. Using fluid and rigid particles of largely different sizes as, e.g., in Winchenbach et al. (2017), is definitely an interesting opportunity for future research as it would decouple the accuracy of the fluid and the rigid-body solver. This, however, might induce adjustments in the fluid solver, e.g., adaptive kernel supports, which is currently not required in our coupling approach. Another consequence of using particles for the rigid-rigid contact handling is that their size and velocity influence the maximum allowed time step due to the CFL condition. However, this time-step restriction generally applies for particle-based boundary handling when the fluid is in contact with a moving rigid body. Furthermore, in our scenes, the CFL condition of the fluid solver normally dominated the time step.

As already illustrated in Section 4.1, the performance gain factor varies and depends on the scenario. Our approach is especially useful in scenes with large velocities or large accelerations at particles. It is also advantageous in scenes with large density ratios. In the rising sphere scenario in Figure 2, e.g., we use a moderate density ratio of 1:10. If we would use a larger ratio, then we could present larger speedups. However, the performance gain is reduced in scenarios with smaller density ratios and also in calm scenes with slowly moving particles.

6 CONCLUSION

We have introduced a strong two-way coupling method for SPH fluids and rigid bodies with particle-sampled surfaces.

On one hand, the technique shares characteristics of a unified approach by solving pressure at fluid and rigid-body particles based on SPH density deviations and also by deriving fluid-rigid and rigid-rigid contact forces from pressure using the SPH methodology. Further, the proposed rigid-body solver is closely interconnected with the fluid solver. Each fluid solver iteration influences the rigid-body solver and each iteration of the rigid-body solver affects the fluid solver.

On the other hand, although we solve a monolithic system, we show that it is a flexible and useful extension for iterative SPH pressure solvers in general. This is emphasized by the description of the coupling concept, where we specify the interface between fluid and rigid-body solver, but keep the fluid solver in a general format. The flexibility is further shown in the experiments, where we combine our coupling with DFSPH, IISPH, highly viscous fluids, and elastic solids. Large scenarios with up to $90M$ particles have been presented.

Our two-way coupling stabilizes the fluid-rigid interface handling. This is illustrated in comparisons with the approach of Akinci et al. (2012). This stabilization enables larger simulation

time steps, which corresponds to significant performance improvements of up to a factor of 58. The introduced SPH-based rigid-rigid contact handling does not only promote the two-way coupling, but is also able to handle complex contact geometries between rigid objects with tens of thousands of simultaneous contacts.

In the future, we plan to investigate the applicability of our concept to alternative Lagrangian fluid formulations, i.e., position-based fluids (Macklin and Müller 2013) due to the versatility of the concept and power particles (de Goes et al. 2015) due to the improved accuracy compared to SPH interpolation. Both approaches can handle particle sampled boundaries, but the interface to our two-way coupling has to be investigated. In our current setting, the fluid solver provides fluid-rigid interface forces to the rigid bodies. Further, the fluid solver has to be able to process velocity changes at rigid-body particles during the iterations. Additionally, it could be interesting to combine the proposed strong coupling with alternative particle-based rigid-body solvers (e.g., solvers based on DEM as proposed by Bell et al. (2005) or Harada (2007)).

APPENDIX

A CHOICE OF γ

We use a value of 0.7 for γ for the computation of the artificial rest volume of a rigid particle in Section 3.4.2. This follows from the following observation. If a volume is uniformly sampled with particles at distance h, then we get $\frac{1}{\sum_k W_{rk}} = h^3$ for a particle r with complete neighborhood and neighbors k, i.e., its volume h^3 is correctly computed from the inverse of the number density. If we remove particles from the neighborhood of r, then its volume would be overestimated. These missing contributions can be compensated by γ. If 3D particles are uniformly sampled in a 2D plane, which we assume to be the case in our rigid-body representation, then the desired volume would still be h^3, but we get $\frac{1}{\sum_k W_{rk}} \approx \frac{h^3}{0.7}$. Computing $\frac{0.7}{\sum_k W_{rk}}$ instead, i.e., $\gamma = 0.7$, results in the desired volume h^3.

The effect of using different values for γ is demonstrated in Figure 13.

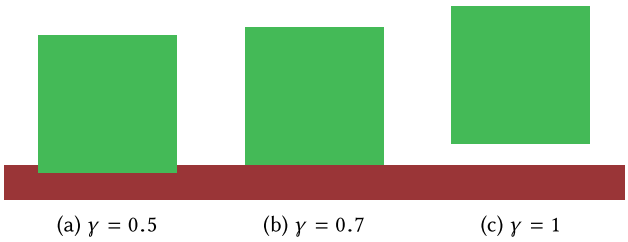

Fig. 13. The effect of using different values of γ for the computation of the artificial rest volume of a rigid particle. The green cube falls onto the static red plane. If γ is chosen lower than 0.7, then a collision is only detected and resolved after the two objects already overlap. When choosing $\gamma = 1$, the two rigid bodies come to rest without touching each other. $\gamma = 0.7$ gives the desired result.

ACKNOWLEDGMENTS

The rubber duck by willie is licensed under CC0 1.0. The rubber duck mold by rocketboy is licensed under CC BY-NC 3.0. The nut and bolt models by quezz38 are licensed under CC BY 3.0. The propeller model by Parkinbot is licensed under CC BY-NC 3.0. The Armadillo model is courtesy of the Stanford University Computer Graphics Laboratory. The Orion capsule is courtesy of NASA. The creature model is courtesy of Eric Mootz. The tree model is from www.cadnav.com. The rock models by Andrew Hansen are licensed under CC0 1.0. We thank FIFTY2 Technology's team members for their support.

REFERENCES

S. Adami, X. Y. Hu, and N. A. Adams. 2012. A generalized wall boundary condition for smoothed particle hydrodynamics. *J. Comput. Phys.* 231, 21 (2012), 7057–7075.
Muzaffer Akbay, Nicholas Nobles, Victor Zordan, and Tamar Shinar. 2018. An extended partitioned method for conservative solid-fluid coupling. *ACM Trans. Graph.* 37, 4 (2018), 86:1–86:12.
Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2013a. Versatile surface tension and adhesion for SPH fluids. *ACM Trans. Graph.* 32, 6 (2013), 182:1–182:8.
Nadir Akinci, Jens Cornelis, Gizem Akinci, and Matthias Teschner. 2013b. Coupling elastic solids with smoothed particle hydrodynamics fluids. *Comput. Animat. Virt. 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 Trans. Graph.* 31, 4 (2012), 62:1–62:8.
Ryoichi Ando, Nils Thürey, and Chris Wojtan. 2015. A stream function solver for liquid simulations. *ACM Trans. Graph.* 34, 4 (2015), 53:1–53:9.
Stefan Band, Christoph Gissler, Markus Ihmsen, Jens Cornelis, Andreas Peer, and Matthias Teschner. 2018a. Pressure boundaries for implicit incompressible SPH. *ACM Trans. Graph.* 37, 2 (Feb. 2018), 14:1–14:11.
Stefan Band, Christoph Gissler, Andreas Peer, and Matthias Teschner. 2018b. MLS pressure boundaries for divergence-free and viscous SPH fluids. *Comput. Graph.* 76 (2018), 37–46.
Stefan Band, Christoph Gissler, and Matthias Teschner. 2017. Moving least squares boundaries for SPH fluids. In *Virtual Reality Interactions and Physical Simulations*. The Eurographics Association.
Christopher Batty, Florence Bertails, and Robert Bridson. 2007. A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph.* 26, 3 (2007).
Markus Becker, Hendrik Tessendorf, and Matthias Teschner. 2009. Direct forcing for lagrangian rigid-fluid coupling. *IEEE Trans. Visual. Comput. Graph.* 15, 3 (2009), 493–503.
Nathan Bell, Yizhou Yu, and Peter J. Mucha. 2005. Particle-based simulation of granular materials. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 77–86.
Jan Bender, Kenny Erleben, and Jeff Trinkle. 2014. Interactive simulation of rigid body dynamics in computer graphics. *Comput. Graph. Forum* 33, 1 (2014), 246–270.
Jan Bender and Dan Koschier. 2017. Divergence-free SPH for incompressible and viscous fluids. *IEEE Trans. Visual. Comput. Graph.* 23, 3 (2017), 1193–1206.
Jan Bender, Dan Koschier, Tassilo Kugelstadt, and Marcel Weiler. 2017. A micropolar material model for turbulent SPH fluids. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 4:1–4:8.
Mark Carlson, Peter J. Mucha, and Greg Turk. 2004. Rigid fluid: Animating the interplay between rigid bodies and fluid. *ACM Trans. Graph.* 23, 3 (2004), 377–384.
Nuttapong Chentanez, Tolga G. Goktekin, Bryan E. Feldman, and James F. O'Brien. 2006. Simultaneous coupling of fluids and deformable bodies. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 83–89.
Nuttapong Chentanez and Matthias Müller. 2010. Real-time simulation of large bodies of water with small scale details. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 197–206.
Simon Clavet, Philippe Beaudoin, and Pierre Poulin. 2005. Particle-based viscoelastic fluid simulation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 219–228.
Erwin Coumans. 2018. Bullet physics library. Retrieved from http://bulletphysics.org/.
Fernando de Goes, Corentin Wallez, Jin Huang, Dmitry Pavlov, and Mathieu Desbrun. 2015. Power particles: An incompressible fluid solver based on power diagrams. *ACM Trans. Graph.* 34, 4 (2015), 50–1.
Mathieu Desbrun, Marie-Paule Cani, et al. 1996. Smoothed particles: A new paradigm for animating highly deformable bodies. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation*, Vol. 96. Springer, 61–76.

Crispin Deul, Patrick Charrier, and Jan Bender. 2014. Position-based rigid body dynamics. *Comput. Animat. Virt. Worlds* 27, 2 (2014), 103–112.
R. Elliot English, Linhai Qiu, Yue Yu, and Ronald Fedkiw. 2013. Chimera grids for water simulation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 85–94.
Makoto Fujisawa and Kenjiro T. Miura. 2015. An efficient boundary handling with a modified density calculation for SPH. *Comput. Graph. Forum* 34, 7 (2015), 155–162. Retrieved from arXiv:https://onlinelibrary.wiley.com/.
Dan Gerszewski, Ladislav Kavan, Peter-Pike Sloan, and Adam W. Bargteil. 2015. Basis enrichment and solid-fluid coupling for model-reduced fluid simulation. *Comput. Animat. Virt. Worlds* 26, 2 (2015), 109–117.
Robert A. Gingold and Joseph J. Monaghan. 1977. Smoothed particle hydrodynamics: Theory and application to non-spherical stars. *Month. Notices Roy. Astron. Soc.* 181, 3 (1977), 375–389.
Christoph Gissler, Stefan Band, Andreas Peer, Markus Ihmsen, and Matthias Teschner. 2017. Generalized drag force for particle-based simulations. *Comput. Graph.* 69 (2017), 1–11.
Jón Tómas Grétarsson, Nipun Kwatra, and Ronald Fedkiw. 2011. Numerically stable fluid-structure interactions between compressible flow and solid structures. *J. Comput. Phys.* 230, 8 (2011), 3062–3084.
Eran Guendelman, Andrew Selle, Frank Losasso, and Ronald Fedkiw. 2005. Coupling water and smoke to thin deformable and rigid shells. *ACM Trans. Graph.* 24, 3 (2005), 973–981.
Takahiro Harada. 2007. Real-time rigid body simulation on GPUs. In *GPU Gems 3*, Hubert Nguyen (Ed.). Addison-Wesley Professional, Chapter 29, 611–632.
Xiaowei He, Ning Liu, Sheng Li, Hongan Wang, and Guoping Wang. 2012. Local poisson SPH for viscous incompressible fluids. *Comput. Graph. Forum* 31, 6 (2012), 1948–1958.
Markus Ihmsen, Jens Cornelis, Barbara Solenthaler, Christopher Horvath, and Matthias Teschner. 2014a. Implicit incompressible SPH. *IEEE Trans. Visual. Comput. Graph.* 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)*. The Eurographics Association.
Richard Keiser, Bart Adams, Philip Dutré, Leonidas Guibas, and Mark Pauly. 2006. *Multiresolution particle-based fluids.* Technical Report 520. Department of Computer Science, ETH Zurich.
Richard Keiser, Bart Adams, Dominique Gasser, Paolo Bazzi, Philip Dutre, and Markus Gross. 2005. A unified Lagrangian approach to solid-fluid animation. In *Proceedings of the Eurographics/IEEE VGTC Symposium Point-Based Graphics*. 125–148.
Bryan M. Klingner, Bryan E. Feldman, Nuttapong Chentanez, and James F. O'Brien. 2006. Fluid animation with dynamic meshes. *ACM Trans. Graph.* 25, 3 (2006), 820–825.
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, 1:1–1:10.
Nipun Kwatra, Chris Wojtan, Mark Carlson, Irfan E. Essa, Peter J. Mucha, and Greg Turk. 2010. Fluid simulation with articulated bodies. *IEEE Trans. Visual. Comput. Graph.* 16, 1 (2010), 70–80.
Michael Lentine, J. T. Gretarsson, Craig Schroeder, Avi Robinson-Mosher, and Ronald Fedkiw. 2011. Creature control in a fluid environment. *IEEE Trans. Visual. Comput. Graph.* 17, 5 (2011), 682–693.
Wenlong Lu, Ning Jin, and Ronald P. Fedkiw. 2016. Two-way coupling of fluids to reduced deformable bodies. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Ladislav Kavan and Chris Wojtan (Eds.). The Eurographics Association.
Leon B. Lucy. 1977. A numerical approach to the testing of the fission hypothesis. *Astron. J.* 82 (1977), 1013–1024.
Miles Macklin and Matthias Müller. 2013. Position based fluids. *ACM Trans. Graph.* 32, 4 (2013), 104.
Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. 2014. Unified particle physics for real-time applications. *ACM Trans. Graph.* 33, 4 (2014), 153:1–153:12.
Brian Vincent Mirtich. 1996. *Impulse-based Dynamic Simulation of Rigid Body Systems.* Ph.D. Dissertation.
Joseph J. Monaghan. 1994. Simulating free surface flows with SPH. *J. Comput. Phys.* 110, 2 (1994), 399–406.
Joseph J. Monaghan. 2005. Smoothed particle hydrodynamics. *Rep. Progr. Phys.* 68, 8 (2005), 1703.
Joseph J. Monaghan. 2012. Smoothed particle hydrodynamics and its diverse applications. *Ann. Rev. Fluid Mech.* 44 (2012), 323–346.
Joseph P. Morris, Patrick J. Fox, and Yi Zhu. 1997. Modeling low reynolds number incompressible flows using SPH. *J. Comput. Phys.* 136, 1 (1997), 214–226.
Matthias Müller, David Charypar, and Markus Gross. 2003. Particle-based fluid simulation for interactive applications. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 154–159.
Matthias Müller, Simon Schirm, Matthias Teschner, Bruno Heidelberger, and Markus Gross. 2004. Interaction of fluids with deformable solids. *Comput. Animat. Virtual Worlds* 15, 3–4 (2004), 159–171.
G. Oger, M. Doring, B. Alessandrini, and P. Ferrant. 2006. Two-dimensional SPH simulations of wedge water entries. *J. Comput. Phys.* 213, 2 (2006), 803–822.
Seungtaik Oh, Younghee Kim, and Byung-Seok Roh. 2009. Impulse-based rigid body interaction in SPH. *Comput. Animat. Virtual Worlds* 20, 2–3 (2009), 215–224.
Saket Patkar, Mridul Aanjaneya, Wenlong Lu, Michael Lentine, and Ronald Fedkiw. 2016. Towards positivity preservation for monolithic two-way solid-fluid coupling. *J. Comput. Phys.* 312 (2016), 82–114.
Andreas Peer, Christoph Gissler, Stefan Band, and Matthias Teschner. 2018. An implicit SPH formulation for incompressible linearly elastic solids. *Comput. Graph. Forum* 37, 6 (2018), 135–148. Retrieved from arXiv:https://onlinelibrary.wiley.com/.
Andreas Peer, Markus Ihmsen, Jens Cornelis, and Matthias Teschner. 2015. An implicit viscosity formulation for SPH fluids. *ACM Trans. Graph.* 34, 4 (2015), 114:1–114:10.
Daniel J. Price. 2012. Smoothed particle hydrodynamics and magnetohydrodynamics. *J. Comput. Phys.* 231, 3 (2012), 759–794.
Avi Robinson-Mosher, R. Elliot English, and Ronald Fedkiw. 2009. Accurate tangential velocities for solid fluid coupling. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 227–236.
Avi Robinson-Mosher, Craig Schroeder, and Ronald Fedkiw. 2011. A symmetric positive definite formulation for monolithic fluid structure interaction. *J. Comput. Phys.* 230, 4 (2011), 1547–1566.
Avi Robinson-Mosher, Tamar Shinar, Jon Gretarsson, Jonathan Su, and Ronald Fedkiw. 2008. Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Trans. Graph.* 27, 3 (2008), 46:1–46:9.
Hagit Schechter and Robert Bridson. 2012. Ghost SPH for animating water. *ACM Trans. Graph.* 31, 4 (2012), 61.
Barbara Solenthaler, Peter Bucher, Nuttapong Chentanez, Matthias Müller, and Markus Gross. 2011. SPH based shallow water simulation. In *Virtual Reality Interactions and Physical Simulations*. Eurographics Association.
Barbara Solenthaler and Renato Pajarola. 2008. Density contrast SPH interfaces. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 211–218.
Barbara Solenthaler and Renato Pajarola. 2009. Predictive-corrective incompressible SPH. *ACM Trans. Graph.* 28, 3 (2009), 40:1–40:6.
Barbara Solenthaler, Jürg Schläfli, and Renato Pajarola. 2007. A unified particle model for fluid-solid interactions. *Comput. Animat. Virtual Worlds* 18, 1 (2007), 69–82.
Jos Stam and Eugene Fiume. 1995. Depicting fire and other gaseous phenomena using diffusion processes. In *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques*. ACM, 129–136.
Tetsuya Takahashi, Yoshinori Dobashi, Tomoyuki Nishita, and Ming C. Lin. 2017. An efficient hybrid incompressible SPH solver with interface handling for boundary conditions. *Comput. Graph. Forum* (2017), 1–12.
Tetsuya Takahashi and Ming C. Lin. 2016. A multilevel SPH solver with unified solid boundary handling. In *Pacific Graphics*. Eurographics Association, 517–526.
Jie Tan, Yuting Gu, Greg Turk, and C. Karen Liu. 2011. Articulated swimming creatures. *ACM Trans. Graph.* 30, 4 (2011), 58:1–58:12.
Nils Thürey, Klaus Iglberger, and Ulrich Rüde. 2006. Free surface flows with moving and deforming objects with LBM. In *Vision, Modeling, and Visualization*. Akademische Verlagsgesellschaft Aka GmbH, 193–200.
Nils Thürey, Matthias Müller-Fischer, Simon Schirm, and Markus Gross. 2007. Real-time breaking waves for shallow water simulations. In *Pacific Graphics*. IEEE, 39–46.
Richard Tonge, Feodor Benevolenski, and Andrey Voroshilov. 2012. Mass splitting for jitter-free parallel rigid body simulation. *ACM Trans. Graph.* 31, 4 (2012), 105:1–105:8.
Mauricio Vines, Ben Houston, Jochen Lang, and Won-Sook Lee. 2014. Vortical inviscid flows with two-way solid-fluid coupling. *IEEE Trans. Visual. Comput. Graph.* 20, 2 (2014), 303–315.
Rene Winchenbach, Hendrik Hochstetter, and Andreas Kolb. 2017. Infinite continuous adaptivity for incompressible SPH. *ACM Trans. Graph.* 36, 4 (2017), 102:1–102:10.
Hongyi Xu, Yili Zhao, and Jernej Barbic. 2014. Implicit multibody penalty-based distributed contact. *IEEE Trans. Visual. Comput. Graph.* 20, 9 (2014), 1266–1279.
X. Yan, C-F. Li, X-S. Chen, and S-M. Hu. 2018. MPM simulation of interacting fluids and solids. *Comput. Graph. Forum* 37, 8 (2018), 183–193.
Omar Zarifi and Christopher Batty. 2017. A positive-definite cut-cell method for strong two-way coupling between fluids and deformable bodies. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 7:1–7:11.

Received July 2018; revised September 2018; accepted September 2018