DOI: 10.1111/cgf.12467

COMPUTER GRAPHICS *forum*
Volume 00 (2014), number 0 pp. 1–14

Stable and Fast Fluid–Solid Coupling for Incompressible SPH

X. Shao[1], Z. Zhou[1,*], N. Magnenat-Thalmann[2] and W. Wu[1]

[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China
[2]Institute for Media Innovation, Nanyang Technological University, Singapore

Abstract

The solid boundary handling has been a research focus in physically based fluid animation. In this paper, we propose a novel stable and fast particle method to couple predictive–corrective incompressible smoothed particle hydrodynamics and geometric lattice shape matching (LSM), which animates the visually realistic interaction of fluids and deformable solids allowing larger time steps or velocity differences. By combining the boundary particles sampled from solids with a momentum-conserving velocity-position correction scheme, our approach can alleviate the particle deficiency issues and prevent the penetration artefacts at the fluid–solid interfaces simultaneously. We further simulate the stable deformation and melting of solid objects coupled to smoothed particle hydrodynamics fluids based on a highly extended LSM model. In order to improve the time performance of each time step, we entirely implement the unified particle framework on GPUs using compute unified device architecture. The advantages of our two-way fluid–solid coupling method in computer animation are demonstrated via several virtual scenarios.

Keywords: fluid modelling, point-based animation, physically based animation

ACM CCS: I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism Animation

1. Introduction

Physically based simulations of the fluid motions have been widely used for many applications, such as commercial films and computer games. Among various fluid motions, fluid–solid couplings happen all the time due to the flow characteristics inherent to the fluids. Although great progress has been made in fluid–solid couplings [BBB07, BTT09, KWC*10, AIA*12], especially for the interactions between fluids and deformable objects [MST*04, CGFO06, RMSG*08, ACAT13], certain difficulties still prevail and need to be resolved for particle-based smoothed particle hydrodynamics (SPH) fluids.

First, it is still hard to meet the stability well at the fluid–solid coupling interfaces. To our knowledge, the stable boundary handling of SPH fluids needs to simultaneously address two open issues: the penetration artefacts [MST*04, YLHQ12, ACAT13] under larger time steps or velocity differences, and the particle deficiency issues [SB12, AIA*12] including density discontinuities and particle stacking due to the lack of fluid neighbours. The common penalty force methods [MST*04, HKK07, YLHQ12] prevent penetrations by using stiff boundary forces, but the fluid density and pressure at the interfaces are not estimated correctly. Furthermore, the requirement of large penalty forces for non-penetration restricts the time step. The direct forcing method [BTT09] adopts a predictor–corrector scheme to compute coupling forces and velocities, and guarantees non-penetration under larger time steps, but particle stacking occurs at the interfaces. The particle deficiency issues can be alleviated by considering the contributions of mirror particles [MM97, HA06, SB12] or frozen particles [SSP07, IAGT10, AIA*12] to fluid particles. However, the effectivity of these methods in penetration prevention is determined by the sampling density of mirror particles or frozen particles. Although Akinci *et al.* [ACAT13] adaptively sample the surfaces of deformable objects with relatively contributive boundary particles to prevent undesired leakage, it is hard to determine a suitable sampling density of boundary particles for non-penetrations in the case of larger time steps or velocity differences. To alleviate the particle deficiency issues, we sample the objects with both surface boundary particles (SBPs) and inner boundary particles (IBPs), and consider their relative contributions to the fluid particles in different ways. In combination with a momentum-conserving velocity-position correction scheme suitable for our boundary

*Corresponding author: Z. zhou (zz@vrlab.buaa.edu.cn)

particle sampling, we prevent penetrations under larger time steps or velocity differences.

Secondly, the stable deformation and melting of solid objects coupled to SPH fluids cannot be well handled under the same particle sampling resolution. As for the deformation simulation of SPH solid boundaries, although the previous methods [MST*04, SSP07, YLHQ12] derived from continuum mechanics are physically accurate, they may become instable for large deformations or melting. What's more, these deformation models need much smaller time steps than SPH fluids with the same particle sampling resolution to handle stiff materials, which ultimately limits the time step size of the fluid–solid coupling. Based on a highly extended lattice shape matching (LSM) method [RJ07] in which we update the surface position using a weighted average summation interpolation method and liquify the solid particles in the outside–in way, we present a stable particle model to simulate the deformation and melting of objects coupled to predictive–corrective incompressible SPH (PCISPH) fluids under larger time steps.

Thirdly, the complex calculations of each time step limit interactive or real-time applications of SPH-based fluid–solid coupling. In a practical application, producing high-quality fluid animations requires hundreds of thousands of particles and takes several hours or days to compute a single frame. Although level of detail techniques [APKG07, OK12, SG11] reduce the computational complexity by allocating computing resources to regions with interesting fluid flow behaviours, it is still difficult to achieve high performance for large-scale particle-based fluid simulations. Recently, the growth of the computational power of GPUs has been tremendous, and particle-based SPH methods can easily be integrated with general-purpose computation on GPU (GPGPU) techniques [HKK07, ZSP08 GSS10, YLHQ12]. To accelerate the calculation, enabling interactive simulation with a higher particle resolution, we propose a compute unified device architecture (CUDA)-based parallel algorithm for the entire simulation pipeline.

In this paper, a stable and fast particle method is presented to couple PCISPH-based fluids and LSM-based deformable solids. Specifically, the main contributions of this paper are summarized as follows:

1. We propose a stable particle method to simulate fluid–solid coupling, which allows larger time steps or velocity differences by combining boundary particles with a momentum-conserving velocity-position correction scheme.
2. We simulate the stable deformation and melting of solid objects coupled to PCISPH fluids based on a highly extended LSM method.
3. A CUDA-based parallel algorithm for the entire simulation pipeline is designed to greatly improve the time performance.

2. Related Work

As stated in [IOS*14] that SPH has been employed to model such phenomena as incompressible fluid [BT07, RT09, SP09, HLL*12, BLS12, ICS*14], hairs [HMT01], melting [IUD10], multiphase flow [SP08] and viscoelastic material [CBP05] since Müller *et al.* [MCG03] used it to produce fluid simulations at interactive rates.

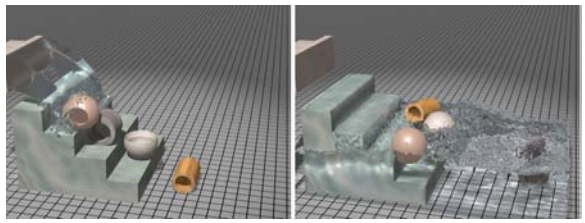

Figure 1: *Interactions of fluid and hollow objects.*

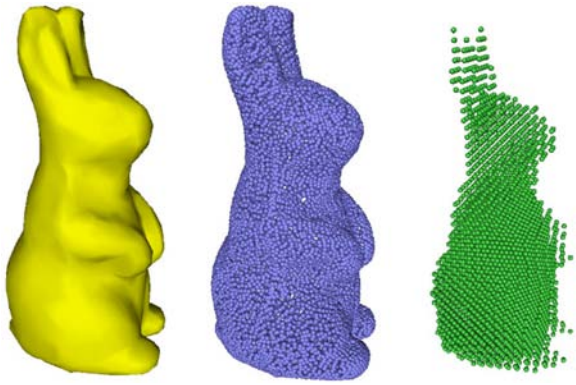

Figure 2: *The representation of solid objects. The triangle mesh (left), surface boundary particles (middle), inner boundary particles (right).*

The more interesting effects, however, emerge when complex solid boundaries are coupled to fluids.

In most SPH-based fluid simulations, boundary conditions are enforced using the penalty forces that scale with the distances of the fluid particles to the solid boundary. Müller *et al.* [MST*04] use the penalty force method to couple SPH fluids to deformable finite element method (FEM) meshes which are sampled with boundary particles. In this case, the stiffness parameter has to be chosen carefully to avoid penetrations and too high pressures. Therefore, small time steps are required to guarantee the stability of the coupling. To overcome this problem, the direct forcing method [BTT09] realizes one- and two-way coupling of fluids and rigid bodies by using predictor–corrector scheme to compute coupling forces and velocities. This method guarantees non-penetration allowing for larger time steps, but stacking of particles leading to irregular density distributions still occurs at the boundaries. Yang *et al.* [YLHQ12] propose a GPU-based real-time method to handle the interaction between SPH fluids and non-linear FEM, in which they combine the direct forcing method with a predictor–corrector scheme to compute the coupling forces. In their method, different boundary conditions and non-penetration robustness can be well guaranteed, but time step limitation and particle stacking artefacts are not well resolved.

The mirror particle method is a more satisfying way to handle the interaction between SPH fluids and solids, because a smoother pressure distribution can be achieved by incorporating dynamically generated mirror particles into the fluid density computation. Hu and Adams [HA06] and Morris and Monaghan [MM97] successfully

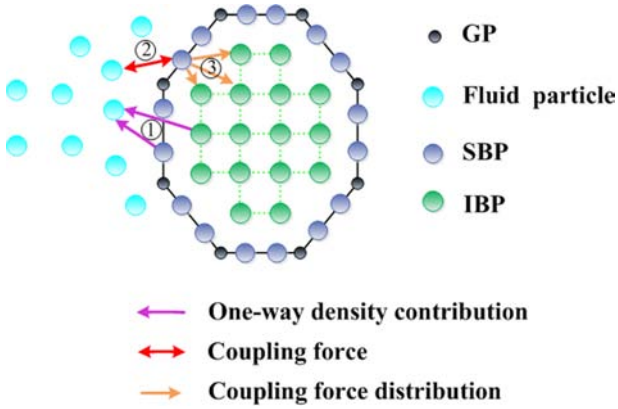

Figure 3: *Three step two-way coupling.*

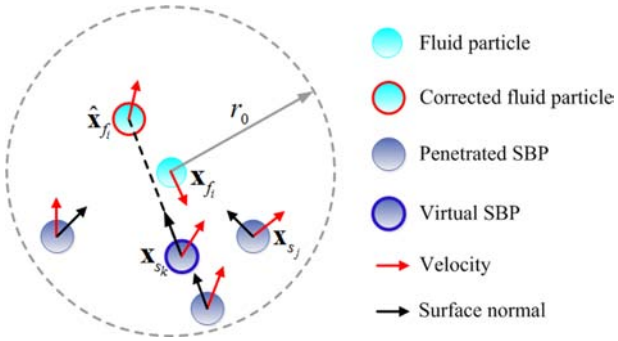

Figure 4: *The velocity-position correction of fluid particle f_i which is considered to penetrate neighbouring SBPs s_j. The red and black arrows represent the velocities and normals of particles.*

apply this approach to handle straight and curved boundary conditions, respectively. Schechter and Bridson [SB12] propose a new particle sampling algorithm to create a narrow layer of mirror particles for the free surface and solid boundary conditions. Their coupling method can handle a spurious numerical surface tension artefact and ensure the mass conservation constraint. However, it is hard to produce mirror particles for deformable solid boundaries.

Solenthaler *et al.* [SSP07] propose a unified particle method to model complex fluid–solid interactions, in which frozen particles are included in density and pressure force computations of fluid. This method keeps the density variation smooth on the boundaries, but the time step has to be chosen small enough to guarantee incompressibility and non-penetration. Ihmsen *et al.* [IAGT10] present a new one-way coupling scheme which combines the direct forcing method and the frozen particle method to handle static boundaries. Therefore, smooth density and pressure distributions are enforced, and larger time steps can be used. Using only one layer of boundary particles, Akinci *et al.* [AIA*12] achieve a momentum-conserving two-way coupling between SPH fluids and complex rigid solids. Their method considers the relative contributions of boundary particles to address the boundary inhomogeneity problem. Based on the above coupling method, Orthmann *et al.* [OHB*13] present a consistent surface model in SPH in conjunction with conservative transport mechanisms within the fluid's surface and between surface and fluid volume, which enables wash-out and coating of rigid objects as well as concentration controlled surface effects like tension, wetting and dragging. By adaptively sampling triangulated surfaces of solids with boundary particles to prevent gaps and undesired leakage, [ACAT13] handles the elastic boundaries of SPH fluids. However, it is difficult to determine the appropriate sampling density of boundary particles to prevent penetrations under larger velocity differences between fluids and solids. In contrast, our method samples the solid body with SBPs and IBPs, and combines these boundary particles with a velocity-position correction scheme to handle the stable fluid-deformable coupling.

In order to further improve the time performance of SPH fluids, several acceleration strategies are presented. Level-of-detail techniques including adaptively sampled [APKG07, OK12] and two-scale [SG11] particle methods have been proposed to improve efficiency by allocating more particles on visually important regions such as near the free surfaces or around objects. Ihmsen *et al.* [IABT11] present an efficient system to simulate SPH fluids by using multiple CPUs in parallel. Recently, the growth of the computational power of GPUs has been tremendous, and SPH model can easily be integrated with GPGPU techniques [HKK07, ZSP08, GSS10, YLHQ12]. We implement our particle-based model entirely on modern GPUs using CUDA, and greatly improve the time performance of fluid–solid coupling simulation.

3. The Coupling Method

3.1. Incompressible SPH (ISPH) fluid solver

In SPH, a field quantity A_i of particle i at position $\mathbf{x}_i$ is calculated as a weighted sum of contributions from all neighbouring particles:

$$A_i = \sum_j m_j \frac{A_j}{\rho_j} W(\mathbf{x}_{ij}, h), \tag{1}$$

where j iterates over all neighbouring particles, $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, m_j is particle mass, ρ_j is the density, A_j is the field quantity at $\mathbf{x}_j$ and W denotes a Gaussian-like kernel with finite support radius h.

Based on the method of [SP08], we compute the density of fluid particle i using

$$\rho_i = m_i \sum_j W(\mathbf{x}_{ij}, h), \tag{2}$$

and calculate the symmetric pressure force $\mathbf{F}_i^p$ and viscosity force $\mathbf{F}_i^v$ exerted on particle i by neighbouring particles using

$$\mathbf{F}_i^p = -\frac{m_i}{\rho_i} \sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W(\mathbf{x}_{ij}, h), \tag{3}$$

$$\mathbf{F}_i^v = \mu \frac{m_i}{\rho_i} \sum_j m_j \frac{\mathbf{v}_{ji}}{\rho_j} \nabla^2 W(\mathbf{x}_{ij}, h), \tag{4}$$

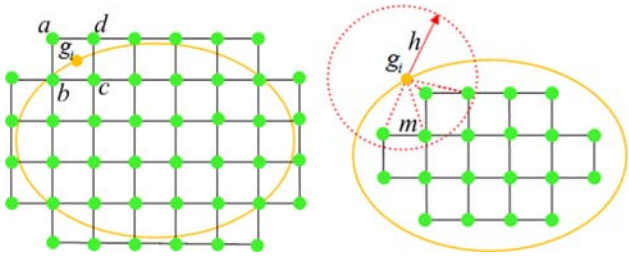

Figure 5: *The deformation model. (a) The original LSM. (b) The extended LSM.*

where $\mathbf{v}$ denotes the velocity, and μ is viscosity coefficient. The pressure p_i of Equation (3) is usually computed by the ideal gas state equation [MCG03 or the Tait equation Mon94, BT07], which suffers from small time steps to enforce fluid incompressibility. In order to use larger time steps, the incompressibility of SPH fluid can be handled by truly ISPH solving a complex pressure Poisson equation (PPE) [RT09, HLL*12]. Especially, Ihmsen *et al.* [ICS*14] propose a discretized form of the PPE that significantly improves the convergence of the solver and the stability of the time-integration scheme.

We enforce the incompressibility of SPH fluids using the promising PCISPH method [SP09], which predicts and corrects the density fluctuation in an iterative manner. In each iteration, the predicted position $\mathbf{x}_i^*(t+\Delta t)$ and velocity $\mathbf{v}_i^*(t+\Delta t)$ of particle i are computed based on $\mathbf{x}_i(t)$, $\mathbf{v}_i(t)$ and the predicted pressure forces. Then, the predicted density $\rho_i^*(t+\Delta t)$ is calculated using the predicted interparticle distance $\mathbf{x}_{ij}^* = \mathbf{x}_i^*(t+\Delta t) - \mathbf{x}_j^*(t+\Delta t)$. Finally, the particle pressure that corrects the predicted density error $\rho_{\mathrm{err}_i}^*(t+\Delta t) = \rho_i^*(t+\Delta t) - \rho_0$ is updated as

$$p_i(t) += \delta \rho_{\mathrm{err}_i}^*(t+\Delta t), \tag{5}$$

where $\delta = \frac{-1}{\beta(-\sum_j \nabla W_{ij} \cdot \sum_j \nabla W_{ij} - \sum_j (\nabla W_{ij} \cdot \nabla W_{ij}))}$. $\beta = \frac{2m_i^2}{\rho_0^2}\Delta t^2$ is a precomputed value, $W_{ij} = W(\mathbf{x}_{ij}, h)$, ρ_0 is the rest fluid density.

To prevent penetrations at the fluid–solid interfaces, we use a velocity-position correction scheme to correct the predicted position when iteratively computing the fluid pressure near the fluid–solid interface, and to correct both velocity and position after the time integration (see Algorithm 1). To improve the time performance, we implement PCISPH entirely on GPUs using CUDA for interactive frame rates (see Section 4).

3.2. Coupling force computations

Our method designs a three-layered particle model to represent the solid boundaries of PCISPH fluids: geometry particles (GPs) for rendering, SBPs used to calculate interactions and IBPs controlling the deformation (see Figure 2). Specifically, GPs, i.e. vertices of the triangle meshes are used to represent the surface of objects for rendering, and to calculate the updated positions and velocities of SBPs via linear interpolation. SBPs deriving from the boundary particle sampling method of [MST*04], contribute to the density and pressure estimations of fluid particles, and exert coupling forces on fluids. Moreover, in our velocity-position correction scheme (see Section 3.3), SBPs are used to detect penetrations and correct the velocities and positions of the penetrated fluid particles. Based on the highly extended LSM method (see Section 3.4), IBPs control the deformation of objects under the coupling forces and other external forces. In order to smooth fluid density distributions at the fluid–solid interfaces, IBPs also contribute to the density estimations of fluid particles. In addition, the positions and velocities of GPs are updated by an SPH interpolation of the values of neighbouring IBPs.

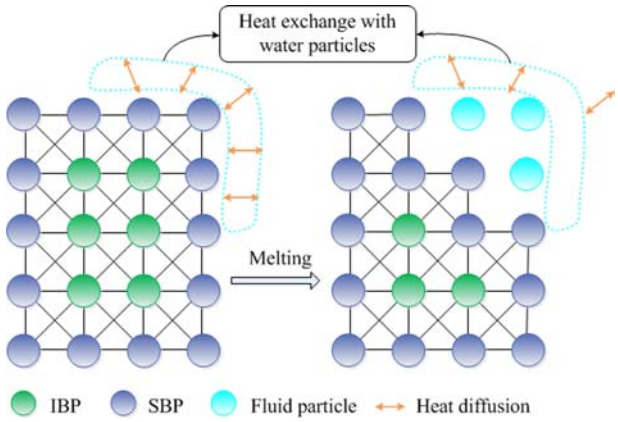

Figure 6: *Topology changes induced by the heat diffusion in the melting.*

Based on the three-layered particle model of solid objects, as shown in Figure 3, we propose a three-step boundary handling algorithm to accurately calculate two-way fluid-solid coupling: (1) Computing one-way density contributions of SBPs and IBPs to fluid particles; (2) Computing two-way coupling forces between SBPs and fluid particles; (3) Distributing the coupling forces exerted on SBPs to neighbouring IBPs. The detailed calculation process is as follows:

Since the adopted kernel function W is spherical, the SPH density summation formulated as Equation (2) underestimates the density of a fluid particle at the fluid–solid interfaces. In order to produce more smooth density and pressure distributions at the fluid–solid interfaces, SBPs and IBPs both serve as frozen particles in our method for the fluid–solid coupling, but are treated in different manners. When calculating the density of a fluid particle near the solid boundary, we take both neighbouring SBPs and IBPs into account. Considering the non-homogeneous distribution of frozen particles due to solid sampling and deformation, we adopt the density model of [AIA*12], which scales the contribution of a boundary particle b_k to the fluid particle f_i by a relative contribution function $\varphi_{b_k}(\rho_0)$

$$\rho_{f_i} = m_{f_i} \sum_j W(\mathbf{x}_{ij}, h) + \sum_k \varphi_{b_k}(\rho_0) W(\mathbf{x}_{ik}, h), \tag{6}$$

where $\varphi_{b_k}(\rho_0) = \rho_0 V_{b_k}$, and V_{b_k} is the current volume of the neighbouring boundary particle b_k.

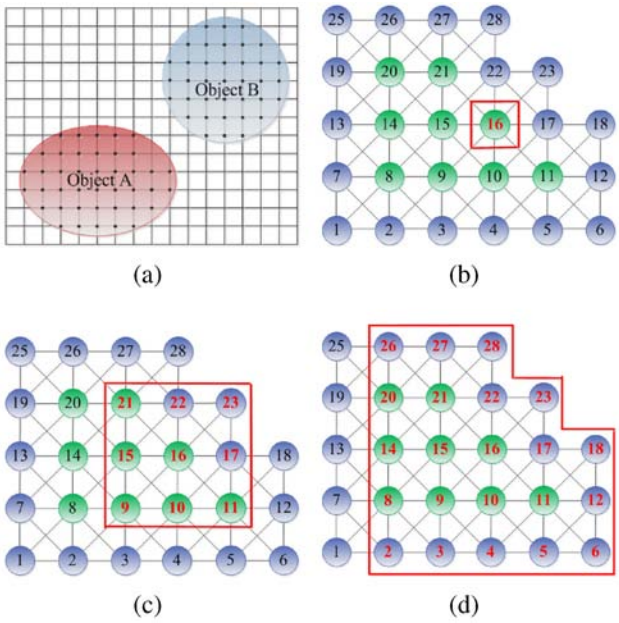

Figure 7: *The iterative filling algorithm of rebuilding regions located on solid particles using CUDA. (a) Voxelizing all objects in the global space. (b)–(d) Filling the 16th region ($w = 2$) through $w + 1$ iterations.*

For solid–fluid coupling, we only consider interactions between fluid particles and SBPs. These coupling forces on SBPs are then propagated to the neighbouring IBPs. Since the fluid–solid coupling forces are only exerted on the interfaces, our approach can handle the coupling accurately. For a pair of coupling particles, fluid particle f_i and SBP s_j, the coupling force $\mathbf{F}^c_{f_i \leftarrow s_j}$ exerted on f_i by s_j includes three components: pressure force $\mathbf{F}^p_{f_i \leftarrow s_j}$, viscosity force $\mathbf{F}^v_{f_i \leftarrow s_j}$ and interface tension $\mathbf{F}^I_{f_i \leftarrow s_j}$.

We adopt the formulation of [AIA*12] to compute the pressure force and viscosity force on a fluid particle f_i exerted by the neighbour SBP s_j:

$$\mathbf{F}^p_{f_i \leftarrow s_j} = -m_{f_i} \varphi_{s_j}(\rho_0) \left(\frac{p_{f_i}}{\rho^2_{f_i}} \right) \nabla W(\mathbf{x}_{ij}, h), \tag{7}$$

$$\mathbf{F}^v_{f_i \leftarrow s_j} = \mu m_{f_i} \varphi_{s_j}(\rho_0) \left(\frac{\mathbf{v}_{ji}}{\rho^2_{s_j}} \right) \nabla^2 W(\mathbf{x}_{ij}, h). \tag{8}$$

The interface tension is responsible for a variety of small-scale fluid phenomena such as the formation of filaments in fluid–solid coupling. Based on the interface tension model of [IUD10], and considering the inhomogeneity of SBPs, we calculate $\mathbf{F}^I_{f_i \leftarrow s_j}$ as follows:

$$\mathbf{F}^I_{f_i \leftarrow s_j} = -\eta_{s_j} m_{f_i} \varphi_{s_j}(\rho_0) W(\mathbf{x}_{ij}, h) \frac{\mathbf{x}_{ij}}{|\mathbf{x}_{ij}|}, \tag{9}$$

Table 1: *Parameter values in the experiments..*

Properties	Values	Unit
Time step (Δt)	0.03	s
Initial spacing (r_0)	0.02	m
Support radius (h)	0.05	m
Density (ρ_0)	100–10 000	kg/m^3
Mass (m)	0.00027–0.027	kg
Viscosity (μ)	0.05	Pa · s
Heat conductivity (K)	0.1–0.8	1
Temperature (T)	−50.0–100.0	°C
Region's half-width (w)	2	1

where η_{s_j} is the interface coefficient of the solid to which the SBP s_j belongs.

And according to [AIA*12], the symmetric coupling force from the fluid particle f_i to the neighbouring SBP s_j is $\mathbf{F}^c_{s_j \leftarrow f_i} = -\mathbf{F}^c_{f_i \leftarrow s_j}$, so we do not need to compute densities and pressures for SBPs.

Finally, our method distributes the coupling force exerted on SBPs to their neighbouring IBPs which control the solid deformation. For an IBP particle d_k, the distributed coupling force is computed by

$$\mathbf{F}_{d_k} = \frac{\sum_j W(\mathbf{x}_{kj}, h) \mathbf{F}^c_{s_j}}{\sum_j W(\mathbf{x}_{kj}, h)}, \tag{10}$$

where s_j denotes neighbouring SBPs.

3.3. Velocity-position correction scheme

The above coupling forces in combination with an appropriate sample density of boundary particles can prevent the penetrations under relative small velocity differences or time steps. However, when the velocity difference or the time step becomes larger, the fluids will penetrate into the object surfaces. To solve these problems, based on [BTT09, YLHQ12], we design a velocity-position correction scheme suitable for our boundary particle sampling and coupling force computation methods. For a pair of coupling particles, a fluid particle f_i and an SBP s_j, the fluid is considered to penetrate the boundary at the position $\mathbf{x}_{s_j}$ if $\mathbf{v}_{ij} \cdot \mathbf{n}_{s_j} < 0$ and $|\mathbf{x}_{ij}| < r_0$. r_0 is the equilibrium distance of fluid particles, and $\mathbf{n}_{s_j}$ denotes surface normal at $\mathbf{x}_{s_j}$. As shown in Figure 4, it is possible that a fluid particle f_i penetrates more than one SBP at the same time in the coupling. So for a fluid particle f_i penetrating several SBPs, we propose to dynamically generate a virtual boundary particle s_k colliding with it. Then the velocity and position of f_i is corrected by simultaneously considering momentum conservation along both normal and tangential directions. Finally, our method distributes the velocity variation of s_k to the neighbouring penetrated SBPs. The detailed correction process is as follows:

A filed quantity $\mathbf{A}_{\mathbf{s_k}}$ of the virtual boundary particle s_k, such as normal, position and velocity, is the weighted average of the values

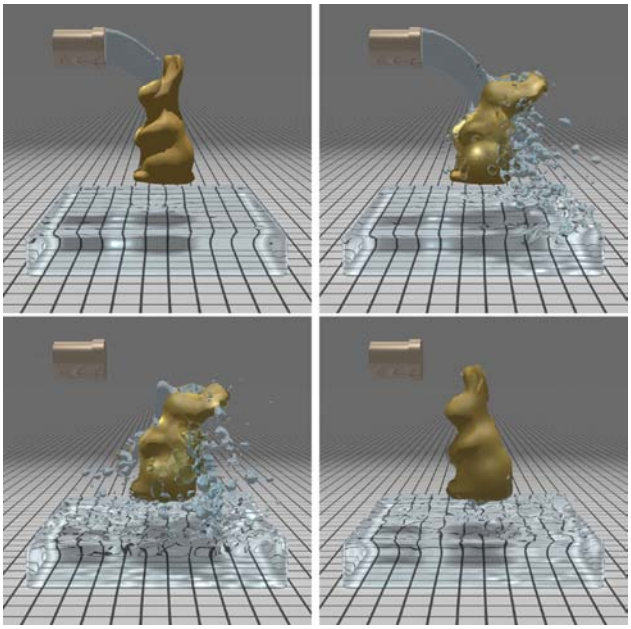

Figure 8: *Fluids poured onto a fixed elastic rabbit.*

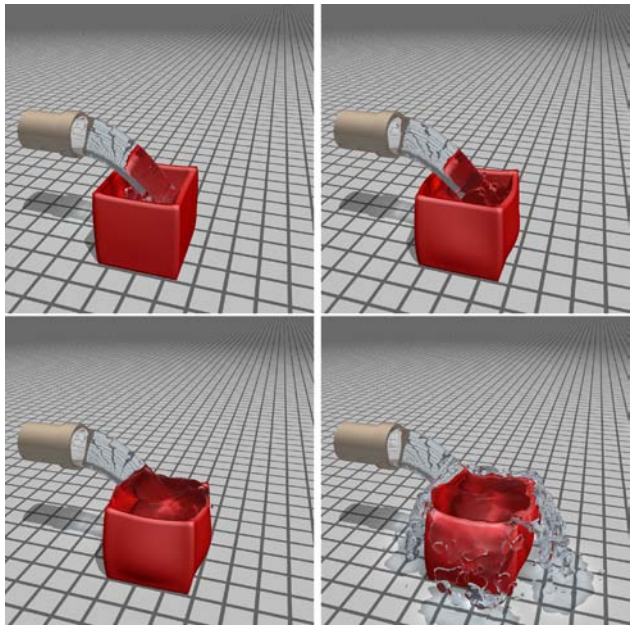

Figure 9: *Fluids poured into an empty elastic box.*

of the neighbouring SBPs penetrated by f_i, which is formulated as:

$$\mathbf{A}_{s_k} = \frac{\sum_j m_{s_j} \mathbf{A}_{s_j} W(\mathbf{x}_{ij}, r_0)}{\sum_j m_{s_j} W(\mathbf{x}_{ij}, r_0)}, \tag{11}$$

where we use $W(\mathbf{x}_{ij}, r_0) = \max(0, (1 - \frac{|\mathbf{x}_{ij}|}{r_0})^3)$ of [YLHQ12].

After the time integration of each time step, by translating the fluid particle f_i along the normal direction $\mathbf{n}_{s_k}$ of the virtual boundary particle s_k, we first correct its position as

$$\hat{\mathbf{x}}_{f_i} = \mathbf{x}_{s_k} + r_0 \mathbf{n}_{s_k}. \tag{12}$$

Then the velocity of f_i is corrected according to boundary material and the law of momentum conservation. We project the velocities of f_i and s_k to the normal direction and the tangential direction of s_k, and get $\mathbf{v}^n_{f_i}$, $\mathbf{v}^t_{f_i}$, $\mathbf{v}^n_{s_k}$ and $\mathbf{v}^t_{s_k}$. Considering momentum conservation along both normal and tangential directions, we formulate the following equations:

$$m_{f_i} \mathbf{v}^n_{f_i} + m_{s_k} \mathbf{v}^n_{s_k} = m_{f_i} \hat{\mathbf{v}}^n_{f_i} + m_{s_k} \hat{\mathbf{v}}^n_{s_k}, \tag{13}$$

$$m_{f_i} \mathbf{v}^t_{f_i} + m_{s_k} \mathbf{v}^t_{s_k} = m_{f_i} \hat{\mathbf{v}}^t_{f_i} + m_{s_k} \hat{\mathbf{v}}^t_{s_k}, \tag{14}$$

where we label a symbol $\wedge$ over the related variables to denote the unknown velocities after the collision.

To enforce the non-penetration constraint at the fluid–solid interfaces, we ensure that the corrected velocity components in the normal direction of s_k are equal, i.e. $\hat{\mathbf{v}}^n_{f_i} = \hat{\mathbf{v}}^n_{s_k}$. Substitute it into Equation (13) and get

$$\hat{\mathbf{v}}^n_{f_i} = \hat{\mathbf{v}}^n_{s_k} = \frac{m_{f_i} \mathbf{v}^n_{f_i} + m_{s_k} \mathbf{v}^n_{s_k}}{m_{f_i} + m_{s_k}}. \tag{15}$$

As for the velocity correction in the tangential direction, we use the method of [YLHQ12] to define a variable ∂ to control the different slip conditions.

$$\partial = \frac{\hat{\mathbf{v}}^t_{f_i} - \hat{\mathbf{v}}^t_{s_k}}{\mathbf{v}^t_{f_i} - \mathbf{v}^t_{s_k}}, \tag{16}$$

where $\partial = 0$ means no-slip in the collision, while $\partial = 1$ states that the collision is free to slip.

Combining Equations (14) and (16), we get

$$\begin{aligned} \hat{\mathbf{v}}^t_{f_i} &= \frac{(m_{f_i} + m_{s_k}\partial)\mathbf{v}^t_{f_i} + m_{s_k}(1-\partial)\mathbf{v}^t_{s_k}}{m_{f_i} + m_{s_k}}, \\ \hat{\mathbf{v}}^t_{s_k} &= \frac{(m_{s_k} + m_{f_i}\partial)\mathbf{v}^t_{s_k} + m_{f_i}(1-\partial)\mathbf{v}^t_{f_i}}{m_{f_i} + m_{s_k}}. \end{aligned} \tag{17}$$

Finally, the velocity variation $\Delta\mathbf{v}_{s_k} = \hat{\mathbf{v}}_{s_k} - \mathbf{v}_{s_k}$ is distributed to the neighbouring penetrated SBPs of fluid particle f_i according to

$$\mathbf{v}_{s_j} += \frac{(\Delta\mathbf{v}_{s_k} \cdot \mathbf{v}_{s_j})\mathbf{v}_{s_j}}{|\mathbf{v}_{s_j}|^2}. \tag{18}$$

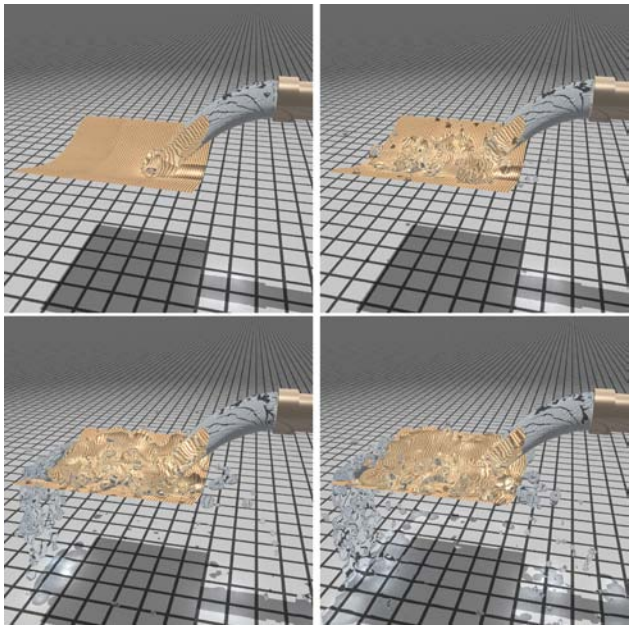

Figure 10: *The coupling of cloth and fluids.*

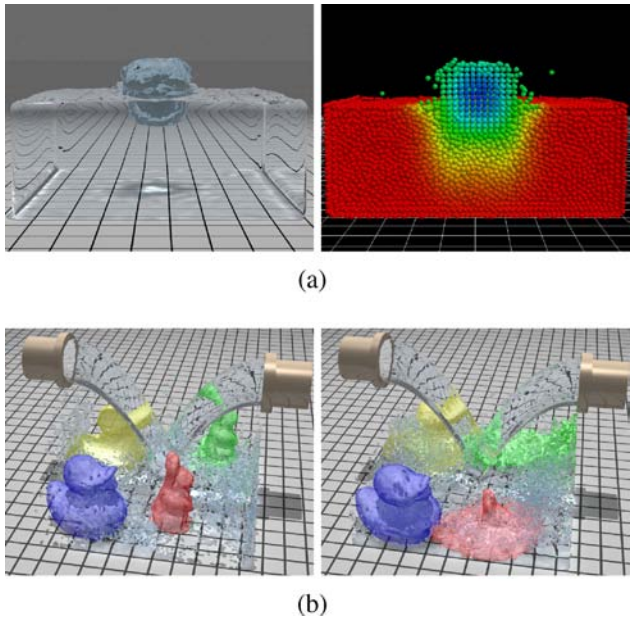

Figure 11: *Melting simulation in the coupling.*

3.4. Solid deformation and melting in the coupling

In this section, we present a stable particle model to simulate the deformation and melting of objects coupled to PCISPH fluids based on LSM method [RJ07].

3.4.1. *Deformation*

The original LSM method [RJ07] divides the particles uniformly sampled from the object's volume into overlapped regions, and calculates the deformation by implementing shape matching on these regions. Then the position of each embedded mesh vertex is computed as the trilinear interpolation of its eight lattice vertices' positions. For example, the deformed position of the mesh vertex GP g_i is the weighted combination of a, b, c and d (see Figure 5a). However, as Figure 5(b) shows, our fluid–solid coupling method only samples the inner of solids with lattice vertices. These inner lattice vertices work as IBPs in the coupling computation, and control the deformation under the distributed forces from SBPs (see Section 3.2). To compute the deformed positions of GPs in our coupling method, we propose a new interpolation method based on the transformation matrices and goal positions of neighbouring IBPs. In the pre-processing stage, for each GP g_i, we build a list of neighbouring IBPs $\{d_j\}$ in the undeformed rest shape. And after implementing shape matching of IBPs in each time step, the deformed position of GP g_i at time t is updated as the weighted average summation interpolation of the goal positions of its neighbouring IBPs:

$$\mathbf{x}_{g_i}^t = \frac{\sum_j W(\mathbf{x}_{ij}^0, h)(\mathbf{R}_{d_j}\mathbf{x}_{ij}^0 + \mathbf{g}_{d_j})}{\sum_j W(\mathbf{x}_{ij}^0, h)}, \tag{19}$$

where $\mathbf{x}_{ij}^0 = \mathbf{x}_{g_i}^0 - \mathbf{x}_{d_j}^0$, $\mathbf{x}_{g_i}^0$ and $\mathbf{x}_{d_j}^0$ are initial positions of GP g_i and neighbouring IBP d_j, respectively, $\mathbf{g}_{d_j}$ denotes the goal position and $\mathbf{R}_{d_j}$ is the average rigid transformation matrix. This interpolation method guarantees that GPs are carried along with the goal positions of initial neighbouring IBPs. Meanwhile, the velocity of GP g_i can be calculated using Equation (1), which is the SPH interpolation of the values of current neighbouring IBPs.

In fluid–solid coupling, due to the requirement of lattice construction in LSM deformation, our method has to adopt at least two layers of IBPs to model thin volumetric objects coupled to PCISPH fluids. But LSM can be extended to model cloth sampled with only one layer of particles. Based on the cluster-based shape matching [SSBT08], we represent the cloth as a regular quadrangular mesh, and then decompose the mesh into overlapping two-dimensional regions. The mesh vertices are selected as both SBPs and IBPs, so the coupling of PCISPH fluid and LSM-based cloth can be handled by our method (see Section 5.1). The fluids couple with two sides of the cloth, so in Figure 4 the normal of SBP s_j penetrated by the fluid particle f_i is replaced with the direction vector $\mathbf{x}_{ij} = \mathbf{x}_{f_i} - \mathbf{x}_{s_j}$.

3.4.2. *Melting*

We propose a stable particle-based model to simulate the melting phenomena in the fluid–solid coupling based on the highly extended LSM. In pre-processing stage, for each lattice vertex particle i, we build a list of one-ring neighbours which share at least one lattice cell with particle i. As shown in Figure 6, the one-ring neighbours are connected by lines. A lattice vertex is labelled as an IBP if it has a maximum number of 27 one-ring neighbours. Otherwise, we label it as an SBP. In the melting simulation, each solid particle i need to be associated with a shape matching region $\Re_i$ which for half-width w contains i and all particles reachable by traversing not

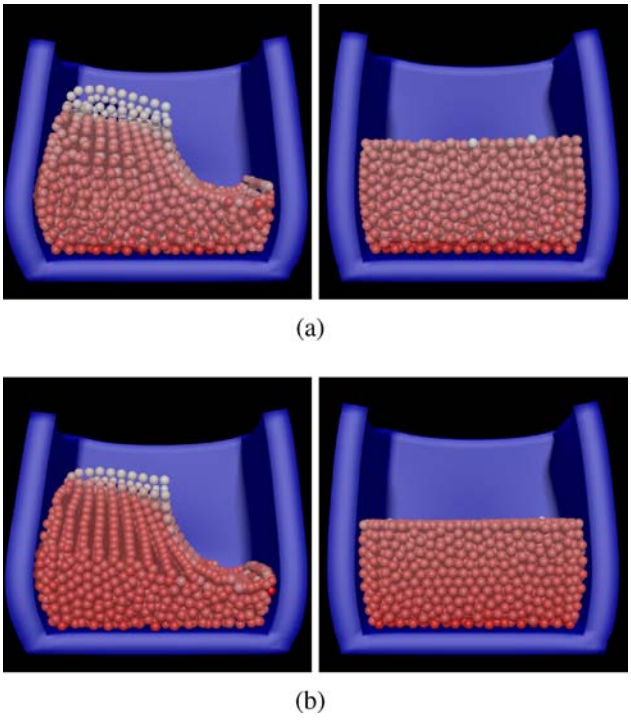

Figure 12: *Pressure profile of an elastic box filled with fluid particles using different methods. (a) [YLHQ12] method; (b) our method.*

more than w lines from particle i. In each time step, the melting process includes two operations: heat diffusion and phase transition.

First, an implicit heat diffusion method is adopted to compute the stable heat transfers. In our method, a temperature parameter value T_i is assigned to each particle, so that the particles carry the physical entity of temperature and transfer it to its neighbouring particles (Figure 6). Since the explicit diffusion method [SSP07, IUD10] is prone to instabilities when the time step becomes too large, we adopt the implicit heat diffusion method [DGP12], and take the heat transfer coefficients of different materials into account. In order to compute the temperature $T_i(t+\Delta t)$ of particle i, T_i is updated for each neighbouring particle by the following operation:

$$T_i \leftarrow T_i(t) + \frac{(K_i + K_j)\Delta t m_j \nabla^2 W_{ij} T_{ji}(t)}{\rho_j + (K_i + K_j)\Delta t \rho_j (\frac{m_i}{\rho_i} + \frac{m_j}{\rho_j})\nabla^2 W_{ij}}, \quad (20)$$

where K_i and K_j denote the coefficients of heat conductivity. According to [DGP12], the above operation is performed twice, and the second time uses the reverse order. The average of the two results is used as temperature $T_i(t+\Delta t)$.

Secondly, we simulate the phase transitions of solid objects using an extended version of LSM. In the melting simulation, each solid particle stores a melting point T_{melt} according to the material characteristics. As shown in Figure 6, when the temperature reaches the melting point, the SBP becomes a liquid particle and separates from its parent object. To model this melting behaviour of solid objects, for each solid particle i, we need dynamically update its one-ring neighbour list and region according to the particles' phases in each time step: First, for each SBP turning into a liquid particle, we directly remove all its one-ring neighbours in the list, and compute its dynamics using PCISPH method; Then, for each remaining solid particle, we update its one-ring neighbour list by removing the liquified particles, and rebuild the regions located on it in parallel (see Section 4). Finally, if an IBP has one-ring neighbours less than 27, it becomes an SBP and gets the coupling forces from the fluid particles. Compared with the melting method [IUD10] in which the inner solid particles may be liquified, our method guarantees that the solid particles are liquified in the outside–in way, which keeps the melting process more stable and realistic.

4. CUDA-Based Implementation

The coupling method we proposed is entirely implemented on the GPU using CUDA for interactive frame rates. Algorithm 1 shows the GPU implementation pipeline.

Algorithm 1: Simulation loop on GPU

```
1  while animating do
2      foreach particle i do
3          Find neighboring particles using kd-tree
4      foreach fluid particle i do
5          Compute viscosity force F_i^v(t) and gravity F_i^g
6          Set pressure p_i(t) = 0
7          Set pressure force F_i^p(t) = (0.0, 0.0, 0.0)^T
8      k=0
9      while k < 3 do
10         foreach fluid particle i do
11             Predict velocity v_i^*(t+Δt), position
               x_i^*(t+Δt)
12             Position correction using Equation (12)
13             Update distances to neighbors
14             Predict density ρ_i^*(t+Δt)
15             Predict density error ρ_{err_i}^*(t+Δt)
16             Update pressure p_i using Equation (5)
17             Compute pressure force F_i^p(t) using
               Equation (7)
18     foreach fluid particle i do
19         Compute F_i^v and F_i^I using Equation (8) and (9)
20     foreach inner boundary particle i do
21         Compute the distributed coupling force using
           Equation (10)
22     foreach solid particle i do
23         Update one-ring neighbor list
24         Rebuild the region located on i
25         Compute goal position g_i using LSM method
26     foreach particle i do
27         Compute the temperature using Equation (20)
28         Compute u_i(t+Δt) and x_i(t+Δt) for IBPs
29         Update GPs' positions using Equation (19)
30     foreach fluid particle i do
31         Velocity and position correction using
           Equation (12), (15) and (17)
32     Surface construction and extraction
```

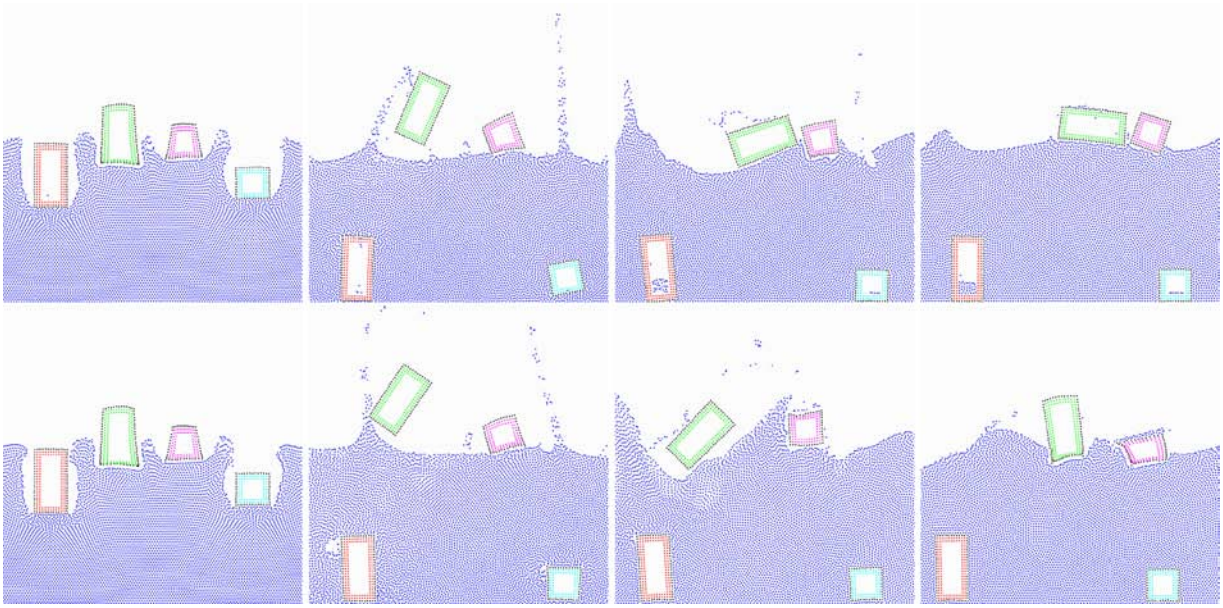

Figure 13: *A 2D coupling scenario without (top) and with (bottom) the velocity-position correction scheme.*

To fully exploit the performance of the GPUs, all unchangeable physical values of particles such as rest positions of IBPs and SBPs, initial neighbouring IBP lists of GPs are stored as textures, while the changeable physical values such as densities, pressures, positions, velocities, forces, goal positions, regions, one-ring neighbours and optimal transformations are stored in GPU global memory. The physical properties of all particles are stored in the same CUDA arrays, and an additional flag value I (fluid particle: 0; SBP: 1 ,..., n; IBP: n+1 ,..., 2n; n is the number of solid objects.) stored in the fourth element of $float4$ is used to distinguish to which object and type each particle belongs. For example, if the flag value of particle i satisfies $I_i > n$ and $I_i\%n == 3$, this particle is an IBP of the third solid object.

We implement PCISPH method [SP09] on GPUs using CUDA to model fluid dynamics and compute the coupling forces exerted on fluid particles. To update the physical values of each particle, the search of neighbouring particles is required, which is also the most time-consuming part in each time step. To speed up this task, based on the method of [ZHWG08], we launch kernels to construct a kd-tree on GPU and traverse it to find neighbouring particles for each particle. For the GPU implementation of the PCISPH method, we first invoke one CUDA thread for each fluid particle to compute the viscosity force (Equation 8) and the interface tension force (Equation 9). Then, another kernel is launched for each fluid particle to compute the pressure (Equation 5) and pressure force (Equation 7) by iteratively predicting and correcting the density fluctuation based on the density model of Equation (6). Different from the original PCISPH [SP09], the predicted positions of fluid particles at the interfaces are corrected (Algorithm 1 line 12) by Equation (12) to prevent penetrations.

Then we enforce the coupling forces on the solid objects. First, we launch a kernel for each fluid particle to compute the coupling force exerted on the neighbouring SBPs. Secondly, another kernel is launched for each IBP to compute the coupling forces distributed by the neighbouring SBPs using Equation (10).

To change the topology of the melting objects on the GPU, we propose an iterative filling algorithm to dynamically update each region according to the phase transition of solid particles (see Figure 7). In pre-processing stage, for each solid particle, we allocate an unsigned integer array *arrayN* of size 27 to store the initial indices of its one-ring neighbours, and an unsigned integer array *arrayR* of size $(2w+1)^3$ to store the initial indices of the particles belonging to the located region. As shown in Figure 7(a), we voxelize all objects in a global space and assign a unique global index to each particle belonging to the objects, which makes it efficient to calculate the initial indices of one-ring neighbours for the running stage. In each time step, the iterative filling operation of $arrayR$ is divided into two steps: First, a kernel is launched for each solid particle to update its one-ring neighbour array $arrayN$ according to the phase transition of solid particles. If a one-ring neighbour turns into a fluid particle, the corresponding index value in $arrayN$ is set to 0. Secondly, we launch another kernel for each region to update $arrayR$. In the kernel, we fill the indices of solid particles belonging to the region (half-width w) into $arrayR$ through $w+1$ iterations. And in each iteration, we fill one-ring neighbours of the solid particles which are already included in the region into $arrayR$. For example, we need to update the region located on the 16th solid particle in Figure 7. In the first iteration, we fill the particle index 16 into $arrayR$ (Figure 7b). And in the second iteration, we look up the array $arrayN$ of the 16th solid particle, and fill its one-ring neighbours into $arrayR$ (Figure 7c). Repeating the same operation in the third iteration, we finally find all solid particles included in the 16th region (Figure 7d). After updating all regions, we calculate the optimal translation and rotation for each region based on the shape matching model, and scatter the averaged goal position of

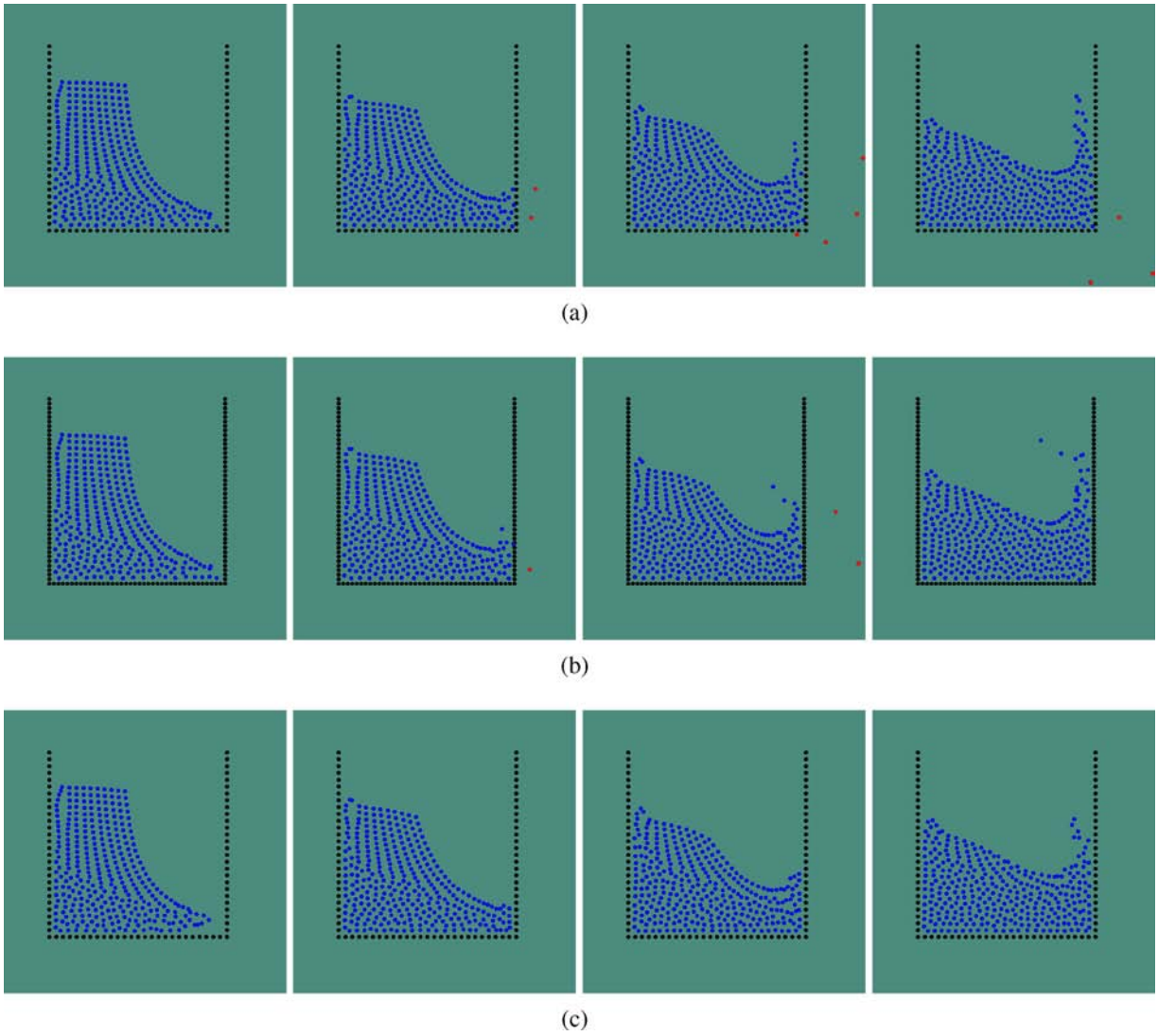

Figure 14: *Comparison with [AIA*12]. (a) [AIA*12] with boundary particle distance r_0. (b) [AIA*12] with boundary particle distance 0.5 * r_0. (c) Our method with boundary particle distance r_0. The red coloured particles are the fluid particles penetrating the box.*

each particle into the global array. Different from the CPU-based fast summation algorithm of [RJ07], the summation of the physical quantities of each region is computed by using a parallel reduction operation.

When the forces and goal positions are computed, we launch a CUDA kernel for each particle to update its position and velocity in different manners according to its type. To prevent penetration artefacts, a kernel is launched for each fluid particle to correct their positions and velocities (Algorithm 1 line 31) according to our velocity-position correction scheme.

To render the surfaces of the melting objects and fluids represented by the particles, we adopt the GPU-based interactive rendering method in [GSS10] to define the distance field constructed from the extracted surface particles. Then the triangle meshes are extracted by using the GPU accelerated marching cube technique provided by the NVIDIA CUDA 'Marching Cubes Isosurfaces' demo.

5. Results and Discussions

We demonstrate the capabilities of our particle-based model for stable and fast fluid–solid coupling via several scenarios. The proposed method is implemented on a PC with an NVIDIA Geforce GTX 690 GPU, Intel Xeon E5630 CPU using c++, CUDA and OpenGL. The images are rendered by using Pov-Ray. The parameter values of the simulation are documented in Table 1.

5.1. Coupling results

Figure 8 is a scenario that a fluid stream ($\rho_0 = 1000$) drops on the surface of an elastic rabbit ($\rho_0 = 5000$) whose bottom is fixed.

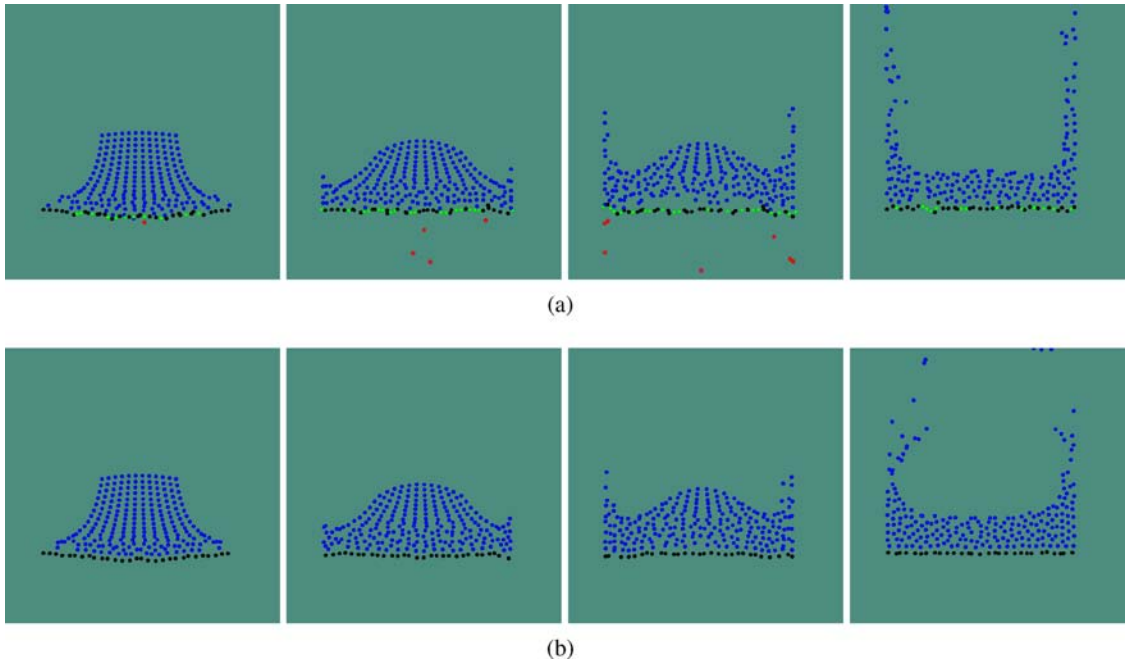

Figure 15: *Comparison with [ACAT13]. (a) [ACAT13] method. (b) Our method. The red particles are the fluid particles penetrating the rod, and the green particles are adaptively sampled boundary particles.*

Table 2: *Time performance (in milliseconds).*

Scene	*#FP*	*#SBP*	*#IBP*	PCISPH	LSM	Coupling	Surface construction	Total
Figure 1	894.6*k*	27.8*k*	51.7*k*	561.5	39.5	21.4	392.8	1015.2
Figure 8	25*k*	3.5*k*	6*k*	17.3	10.1	5.3	15.4	48.1
Figure 9	15*k*	5*k*	3*k*	12.4	7.8	6.3	12.3	38.8
Figure 1 0	15*k*	2.5*k*		12.5	7.0	6.2	12.4	38.1
Figure 1 1(a)	40.5*k*	3.6*k*	16.3*k*	30.2	23.5	10.6	22.7	87.0
Figure 1 1(b)	60*k*	6.2*k*	25.4*k*	46.5	33.4	14.2	35.3	129.4

The maximum magnitude of the velocity ($|\mathbf{v}|_{\max}$) throughout the simulation is 3.35. The fluid stream and the deformable rabbit affect each other, and the flows of fluid on the rabbit surface are realistically simulated by the interface tension.

Our method can handle the coupling of PCISPH fluids ($\rho_0 = 1000$) and thin deformable structures avoiding penetration artefacts. Figure 9 is a scenario that a fluid stream drops into a thin elastic box with density $\rho_0 = 200$ ($|\mathbf{v}|_{\max} = 3.53$). The box consists of one layer of SBPs on the surface, and two layers of IBPs in the middle. Figure 10 shows that a fluid stream drops on a piece of cloth ($\rho_0 = 300$) consisting of only one layer of particles ($|\mathbf{v}|_{\max} = 3.86$). Note that non-penetration can be robustly achieved in these two examples.

Figure 11 demonstrates the ability of our method to simulate the stable and realistic melting of solid objects in the coupling. Figure 11(a) shows the melting of an ice cube with the heat conductivity coefficients ($K = 50$) dropped into the water. The left image shows the realistic rendering result, and the right image visualizes the temperatures of particles. Figure 11(b) shows the melting results of four objects with different heat conductivity coefficients coupling with the fluid streams: green ($K = 70$, $\rho_0 = 200$), red ($K = 50$, $\rho_0 = 10\,000$), yellow ($K = 30$, $\rho_0 = 8000$) and blue ($K = 10$, $\rho_0 = 300$). $|\mathbf{v}|_{\max}$ throughout the simulation is 3.85.

Figure 1 is a scenario that a fluid flow ($\rho_0 = 1000$) at the velocity of 2.0 m/s interacts with several hollow objects [from left to right: ($\rho_0 = 200$, 8000, 300 and 100)] placed on the stair, which demonstrates that our method can simulate the stable coupling of fluid and complex solid objects avoiding penetration artefacts under larger velocity differences (($|\mathbf{v}|_{\max} = 4.78$)).

5.2. Stability analysis

Compared with the method [YLHQ12], our coupling method alleviates the particle deficiency issues at the fluid–solid coupling

interfaces. In Figure 12, we show the pressure profile of an elastic box filled with fluid particles where the front side is clipped to make the fluid visible ($|\mathbf{v}|_{\max} = 3.32$). The fluid particle pressures are colour-coded and proportional to their red saturation. While the method [YLHQ12] leads to pressure noises and particle stacking artefacts (left), our method avoids these problems (right) by taking the relative contributions of boundary particles into account. In addition, the time step of our method ($\Delta t = 0.03$) is much larger than that of [YLHQ12] ($\Delta t = 0.001$), because of the adoption of PCISPH, the extended LSM-based deformation model and the velocity-position correction scheme.

Figure 13 is a 2D scenario that four hollow elastic objects of different densities drop into the fluid without (top) and with (bottom) the velocity-position correction scheme ($|\mathbf{v}|_{\max} = 2.96$). The results show that our velocity-position correction scheme prevents the penetration artefacts for the fluid–solid coupling under larger velocity differences, which increases the reality of the simulation.

Compared with the frozen method [SSP07], the remarkable advantage of our approach is the avoidance of penetration artefacts under larger time steps or velocity differences due to the adoption of the velocity-position correction scheme. Since we only calculate the coupling forces between fluid particles and SBPs sampled on the object surface, our coupling method generates more accurate coupling interfaces. However, our method is not better than [SSP07] on physical accuracy because we adopt the velocity-position correction scheme and the geometrically based LSM model.

Akinci's method [AIA*12] and our method both take the relative contribution of boundary particles to fluid particles, which helps to produce homogenous pressure fields. Different from [AIA*12], our method also take the relative contribution of inner sampled particles (IBPs), which can produce more smoother pressure distribution at the interface. Figure 14 shows an animation scenario where a 2D fluid dam interacts with a rigid box sampled with only one layer of boundary particles ($|\mathbf{v}|_{\max} = 3.54$). Through this animation, we compare our coupling method ($\Delta t = 0.03$) with Akinci's method [AIA*12] ($\Delta t = 0.002$) in the aspect of penetration prevention. When using a low sampling density of boundary particles (Figure 14a), Akinci's method produces penetrations where the velocity difference between fluids and solids is large. And the penetration artefacts are alleviated by using a high sampling density of boundary particles (Figure 14b). Combining the relatively sparse boundary particles with a momentum-conserving correction scheme, our coupling method can prevent penetrations at the interfaces under larger velocity differences (Figure 14c).

In Figure 15, we compare our coupling method with [ACAT13] which adaptively samples triangulated surfaces of solids with boundary particles to prevent gaps. $|\mathbf{v}|_{\max}$ throughout the simulation is 4.13. It is difficult for [ACAT13] to determine an appropriate sampling density of boundary particles to prevent penetrations in the case of very large velocity difference between the fluid and the elastic rod (Figure 15a). Our method avoid penetration artefacts when using relatively sparse boundary particles (Figure 15b). Because LSM cannot simulate the deformation of objects consisting of one line of particles in 2D, the elastic deformation of this example is modelled by meshless shape matching method [MHTG05].

5.3. Time performance

Table 1 shows that our stable coupling method allows for a relative larger time step $\Delta t = 0.03$. Compared with the previous coupling methods [MST*04, YLHQ12], [SSP07] using the same physical parameter values, our approach takes about 30 times larger time steps.

Compared with the coupling methods [SSP07, AIA*12, ACAT13], the time performance of each time step is highly improved by using an entire GPU implementation. Table 2 shows statistics for the average time cost in milliseconds of each simulation step. The number of fluid particles, SBPs and IBPs are noted as *#FP*, *#SBP* and *#IBP*, respectively. The coupling operation includes coupling force computation and distribution. The results demonstrate that the time cost of our method increases with the number of particles, and the melting simulation decreases the performance due to the dynamic update of each region. As shown in Table 2, our method can achieve the interactive simulation of fluid–solid couplings including melting effects. Even for nearly a million particles, the frame rate can be achieved 1.0 *FPS*.

6. Conclusion and Future Work

We have proposed a stable and fast particle method to simulate the two-way interactions of physically based PCISPH fluids and geometric LSM-based deformable objects. By combining the boundary particles sampled from solids with a momentum-conserving velocity-position correction scheme, our method has achieved both the alleviation of the particle deficiency issues and the prevention of the penetrations under larger time steps and velocity differences. The stable deformation and melting of solid objects in the coupling are calculated based on an highly extended LSM method. To improve the time performance, we have entirely implemented the unified particle method on GPUs. Especially, the topology changes of the melting objects on GPUs are achieved by implementing an iterative filling method to update the regions of LSM.

The fluid–solid coupling simulated by our method is visually plausible and stable, which is beneficial to interactive applications such as virtual reality and games. However, the method is not physically completely accurate because the LSM deformation model is unconditionally stable but geometrically motivated. And the velocity-position correction scheme cannot prevent penetrations well when the distance of boundary particles becomes larger than the support radius of boundary particles. Our immediate efforts are geared towards capturing the turbulent details, and simulating complex wetting effects such as wet garments [CMT12] and hairs [RKN12] in the fluid–solid coupling.

Acknowledgements

This work is supported by the National 863 Program of China (Grant no. 2012AA011801), the National Key Technology R&D Program of China (Grant no. 2012BAI06B01) and the National

Natural Science Foundation of China (Grant no. 61472020). We thank the anonymous reviewers for their valuable comments and suggestions which greatly helped to improve the quality of the paper. We also thank Jinsong Zhang for writing the rendering code and helping the video production.

References

[ACAT13] Akinci N., Cornelis J., Akinci G., Teschner M.: Coupling elastic solids with SPH fluids. *Journal of Computer Animation and Virtual Worlds 24*, 3–4 (2013), 195–203.

[AIA*12] Akinci N., Ihmsen M., Akinci G., Solenthaler B., Teschner M.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics 31* (2012), 59–68.

[APKG07] Adams B., Pauly M., Keiser R., Guibas L. J.: Adaptively sampled particle fluids. In *Proceedings of SIGGRAPH* (San Diego, CA, USA, 2007), ACM, pp. 48–54.

[BBB07] Batty C., Bertails F., Bridson R.: A fast variational framework for accurate solid-fluid coupling. In *Proceedings of SIGGRAPH* (San Diego, CA, USA, 2007), ACM, pp. 100–106.

[BLS12] Bodin K., Lacoursiere C., Servin M.: Constraint fluids. *IEEE Transactions on Visualization and Computer Graphics 18*, 3 (March 2012), 516–526.

[BT07] Becker M., Teschner M.: Weakly compressible SPH for free surface flows. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, CA, USA, 2007), ACM, pp. 209–217.

[BTT09] Becker M., Tessendorf H., Teschner M.: Direct forcing for Lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics 15* (2009), 493–503.

[CBP05] Clavet S., Beaudoin P., Poulin P.: Particle-based viscoelastic fluid simulation. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Los Angeles, CA, USA, 2005), ACM, pp. 219–228.

[CGFO06] Chentanez N., Goktekin T. G., Feldman B. E., O'Brien J. F.: Simultaneous coupling of fluids and deformable bodies. In *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Computer Animation* (Vienna, Austria, 2006), ACM, pp. 83–89.

[CMT12] Chen Y. J., Magnenat-Thalmann N.: Physical simulation of wet clothing for virtual humans. *The Visual Computer 28* (2012), 765–774.

[DGP12] Dagenais F., Gagnon J., Paquette E.: A prediction-correction approach for stable SPH fluid simulation from liquid to rigid. In *Proceedings of Computer Graphics International* (Bournemouth, UK, 2012), Springer, pp. 209–217.

[GSS10] Goswami P., Schlegel P., Solenthaler B.: Interactive SPH simulation and rendering on the GPU. In *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Computer Animation* (Madrid, Spain, 2010), ACM, pp. 55–64.

[HA06] Hu X. Y., Adams N. A.: A multi-phase SPH method for macroscopic and mesoscopic flows. *Journal of Computational Physics 213* (2006), 844–861.

[HKK07] Harada T., Koshizuka S., Kawaguchi Y.: Smoothed particle hydrodynamics on GPUs. In *CGI 2007: Proceedings of Computer Graphics International* (Petropolis, Brazil, 2007), Springer, pp. 63–70.

[HLL*12] He X. W., Liu N., Li S., Wang H. G., Wang G. P.: Local poisson SPH for viscous incompressible fluids computer graphics forum. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (EPFL, Switzerland, 2012), ACM, pp. 1948–1958.

[HMT01] Hadap S., Magnenat-Thalmann N.: Modeling dynamic hair as a continuum. In *Proceedings of EUROGRAPHICS* (Manchester, UK, 2001), Wiley-Blackwell, pp. 329–338.

[IABT11] Ihmsen M., Akinci N., Becker M., Teschner M.: A parallel SPH implementation on multi-core CPUs. *Computer Graphics Forum 30* (2011), 99–112.

[IAGT10] Ihmsen M., Akinciand N., Gissler M., Teschner M.: Boundary handling and adaptive time-stepping for PCISPH. In *VRIPHYS 2010: Proceedings of Workshop on Virtual Reality Interaction and Physical Simulation* (Copenhagen, Denmark, 2010), pp. 79–88.

[ICS*14] Ihmsen M., Cornelis J., Solenthaler B., Horvath C., Teschner M.: Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics* 20, 3 (March 2014), 426–435.

[IOS*14] Ihmsen M., Orthmann J., Solenthaler B., Kolb A., Teschner M.: SPH fluids in computer graphics. In *Proceedings of EUROGRAPHICS-State of the Art Reports* (Strasbourg, France, 2014), pp. 21–42.

[IUD10] Iwasaki K., Uchida H., Dobashi Y.: Fast particle based visual simulation of ice melting. In *Proceedings of Pacific Graphics* (Hangzhou, China, 2010), pp. 2215–2223.

[KWC*10] Kwatra N., Wojtan C., Carlson M., Essa I., Mucha P. J., Turk G.: Fluid simulation with articulated bodies. *IEEE Transactions on Visualization and Computer Graphics 16* (2010), 70–80.

[MCG03] Müller M., Charypar D., Gross M.: Particle-based fluid simulation for interactive applications. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, CA, USA, 2003), ACM, pp. 154–159.

[MHTG05] Müller M., Heidelberger B., Teschner M., Gross M.: Meshless deformations based on shape matching. In *Proceedings*

of SIGGRAPH (Los Angeles, CA, USA, 2005), ACM, pp. 471–478.

[MM97] Morris J. P., Monaghan J. J.: A switch to reduce SPH viscosity. *Journal of Computational Physics 136* (1997), 41–50.

[Mon94] Monaghan J.: Simulating free surface flows with SPH. *Journal of Computational Physics 110* (1994), 399–406.

[MST*04] Müller M., Schirm S., Teschner M., Heidelberger B., Gross M.: Interaction of fluids with deformable solids. In *CASA 2004: Proceedings of Computer Animation and Social Agents* (Geneva, Switzerland, 2004), Computer Graphics Society (CGS), pp. 159–171.

[OHB*13] Orthmann J., Hochstetter H., Bader J., Bayraktar S., Kolb A.: Consistent surface model for SPH-based fluid transport. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Anaheim, CA, USA, 2013), ACM, pp. 95–103.

[OK12] Orthmann J., Kolb A.: Temporal blending for adaptive SPH. *Computer Graphics Forum 31* (2012), 2436–2449.

[RJ07] Rivers A. R., James D. L.: FastLSM: Fast lattice shape matching for robust real-time deformation. In *Proceedings of SIGGRAPH* (San Diego, CA, USA, 2007), ACM, pp. 109–116.

[RKN12] Rungjiratananon W., Kanamori Y., Nishita T.: Wetting effects in hair simulation. In *Proceedings of Pacific Graphics* (Hong Kong, China, 2012), Wiley-Blackwell, pp. 319–328.

[RMSG*08] Robinson-Mosher A., Shinar T., Gretarsson J., Su J., Fedkiw R.: Two-way coupling of fluids to rigid and deformable solids and shells. In *Proceedings of SIGGRAPH* (Los Angeles, CA, USA, 2008), ACM, pp. *46*:1–46:9.

[RT09] Rafiee A., Thiagarajan K. P.: An SPH projection method for simulating fluid-hypoelastic structure interaction. *Computer Methods in Applied Mechanics and Engineering 198* (2009), 1948–1958.

[SB12] Schechter H., Bridson R.: Ghost SPH for animating water. In *SIGGRAPH 2012: Proceedings of SIGGRAPH* (Los Angeles, CA, USA, 2012), ACM, pp. 41–50.

[SG11] Solenthaler B., Gross M.: Two-scale particle simulation. In *Proceedings of SIGGRAPH* (Vancouver, Canada, 2011), ACM, pp. 811–818.

[SP08] Solenthaler B., Pajarola R.: Density contrast SPH interfaces. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Dublin, Ireland, 2008), ACM, pp. 211–218.

[SP09] Solenthaler B., Pajarola R.: Predictive-corrective incompressible SPH. In *Proceedings of SIGGRAPH* (New Orleans, LA, USA, 2009), ACM, pp. 40–49.

[SSBT08] Stumpp T., Spillmann J., Becker M., Teschner M.: A geometric deformation model for stable cloth simulation. In *Proceedings of Workshop on Virtual Reality Interaction and Physical Simulation* (Grenoble, France, 2008), pp. 39–46.

[SSP07] Solenthaler B., Schlafli J., Pajarola R.: A unified particle method for fluid-solid interactions. *Computer Animation and Virtual Worlds 18* (2007), 69–82.

[YLHQ12] Yang L. P., Li S., Hao A. M., Qin H.: Realtime two-way coupling of meshless fluids and nonlinear FEM. *Computer Graphics Forum 31*, (2012), 2037–2046.

[ZHWG08] Zhou K., Hou Q. M., Wang R., Guo B. N.: Real-time kd-tree construction on graphics hardware. In *Proceedings of SIGGRAPH* (Los Angeles, CA, USA, 2008), ACM, pp. 209–217.

[ZSP08] Zhang Y., Solenthaler B., Pajarola R.: Adaptive sampling and rendering of fluids on the GPU. In *Proceedings of Symposium on Point-Based Graphics* (Los Angeles, CA, USA, 2008), pp. 137–146.