Local Poisson SPH For Viscous Incompressible Fluids

Xiaowei He[1,2], Ning Liu[1], Sheng Li[1], Hongan Wang[2] and Guoping Wang[1]

[1]Graphics Lab of EECS, Peking University, China
[2]State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China
{hexw, liuning, lisheng}@graphics.pku.edu.cn, hongan.w@gmail.com, wgp@pku.edu.cn

Abstract

Enforcing fluid incompressibility is one of the time-consuming aspects in SPH. In this paper, we present a local Poisson SPH (LPSPH) method to solve incompressibility for particle based fluid simulation. Considering the pressure Poisson equation, we first convert it into an integral form, and then apply a discretization to convert the continuous integral equation to a discretized summation over all the particles in the local pressure integration domain determined by the local geometry. To control the approximation error, we further integrate our local pressure solver into the predictive-corrective framework to avoid the computational cost of solving a pressure Poisson equation globally. Our method can effectively eliminate the large density deviations mainly caused by the solid boundary treatment and free surface topological change, and show advantage of a higher convergence rate over the predictive-corrective incompressible SPH (PCISPH).

Keywords: Local Poisson SPH, fluid simulation, incompressibility

ACM CCS: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and RealismAnimation

1. Introduction

Fluid simulation plays an important role in the entertainment and engineering fields, it aims to reproduce the real flow state based on hydrodynamics and tries to give us a totally immersive experience. Since Smoothed Particle Hydrodynamics (SPH) was first applied to interactive fluid simulation by [MCG03], many studies [BT07, APKG07, SP09] have been done on this fully Lagrange meshless method to get an efficient and realistic simulation result. However, one main drawback of SPH is its high computational cost to enforce incompressibility which restricts the simulation scale to get photo-realistic fluid animations. In the commonly used *weakly compressible* SPH (WCSPH)[Mon94, BT07], small time steps are required to reduce the density fluctuation, which increases the overall computation cost in simulating water. Alternatively, *predictive-corrective incompressible* SPH (PCISPH) [SP09] allows for larger time steps. However, compared to WCSPH, more computation cost is required for one simulation step due to the iteration scheme.

In this paper, we present a new *local Poisson* SPH (LPSPH) method to solve incompressibility, which retains both the high computing efficiency per physical step of WCSPH and the large time step of truly *incompressible* SPH (ISPH). Based on partial differential equation theories [Eva98], we convert the pressure Poisson equation [CR99] to a discretized summation over all the particles in the local pressure integration domain. Then we can simply compute each particle's pressure by summation operations. To control the approximation error, we finally integrate our local pressure solver into the predictive-corrective framework defined in [SP09] to achieve a strict density error control.

Unlike previous methods of ISPH, our algorithm can avoid the computational cost of solving a pressure Poisson equation globally, and effectively eliminate the large density deviations caused by the solid boundary treatment and free surface topological change. Compared to WCSPH, we can get a higher rate of density error decay. After integrating the predictive-corrective framework, the resulted *iterative*-LPSPH also shows higher convergence rate over the

predictive-corrective incompressible SPH (PCISPH). Nevertheless, it is still worth to mention that the neighbourhood query is a key factor which limits the efficiency of enforcing the fluid incompressibility [IABT11].

The rest of this paper is composed as follows: Section 2 reviews the related work to our method, Section 3 introduces some basic theories of fluid dynamics and SPH techniques, Section 4 describes the details of our local Poisson SPH method. Then in Section 5, several examples are presented to compare our method with WCSPH and PCISPH. Finally, we draw a conclusion and discuss our future work in Section 6.

2. Related Work

SPH was originally developed to model the astrosphere interactions in astronautics by the pioneer work [Luc77, GM77]. The early work [DG96] first introduced the SPH method to graphics community. They proposed proper kernels for computing forces between particles to simulate deformable solids. Since then, particle-based methods such as SPH have become an active topic as the meshless characteristic makes it very suitable to simulate fast topology-changing phenomena such as fluid motion. [MCG03] first applied the SPH method to fluid simulation in 2003. They proposed several kernels to compute particle densities, pressure forces and viscous forces. Later, more techniques were proposed including adaptive fluids [APKG07], fluid-solid interaction [MSKG05, KAG*05], melting [SSP07] and porous media [LAD08, LD09].

Recently, research is focused on developing incompressible fluid solver using particle-based method. The fluid incompressibility is from the Navier-Stokes equations and need to be solved to conserve volume. When modelling incompressible flows with SPH, two common strategies have been used: the weakly compressible SPH (WCSPH) method and truly incompressible SPH (ISPH). WCSPH models pressure from a stiff equation of state (EOS) [Mon94, MFZ97] related to the speed of the acoustic waves. However, to meet the Courant-Friedrichs-Levy (CFL) condition [CFL67], it usually requires a very small time step associated with a speed of the acoustic at least 10 times higher than the maximum of velocity. Besides, it is always cumbersome to choose an appropriate parameter of the EOS equation. These two drawbacks make it infeasible to simulate large-scale fluid within reasonable time. In contrast to WCSPH, ISPH needs to solve a pressure poisson equation to project the velocity field to a divergence-free field. To achieve this, [CR99] first integrated the velocity field forward without enforcing incompressibility, the resulting intermediate velocity field was then projected onto a divergence-free space by solving a pressure Poisson equation derived from an approximate pressure projection. [PTB*03] realized incompressibility by Moving Particles Semi-implicit (MPS) method, which was ca-

pable of solving a variety of incompressible fluids. [LKO05, HA07, LTKF08] enforced both a divergence-free velocity field and a targeting of particle number density. However, the problem of ISPH lies in the complexity to solve the pressure Poisson equation globally, which causes much higher computational cost in a single simulation step compared to WCSPH.

Some other efforts have also been made. [SBH09] proposed a novel method to handle point-based incompressibility. They used Voronoi cells to segment the velocity field and ensured that each cell is divergence-free. [BZZW09] presented a revised SPH method to avoid solving pressure Poisson equation and achieved more efficient simulation. [SP09] gave a scheme called predictive-corrective incompressible SPH(PCISPH). Compared to WCSPH, PCISPH eliminated the costly procedure of solving the pressure Poisson equation with an iterative pressure correction algorithm, which can also enlarge the time step size to achieve better simulation performances. Moreover, [BLS11] enforced incompressibility and boundary conditions by using holonomic kinematic constraints on the density and [RWT11] presented a hybrid method that enforced a divergence free velocity on a coarse grid accompanied by a local density correction of the particles.

3. Fundamentals of Fluid Dynamics

The governing equations for simulating free surface flow are the mass and momentum conservation laws. Regarding to the Lagrangian frame, they are written as:

$$\frac{1}{\rho}\frac{D\rho}{Dt} + \nabla \cdot \mathbf{v} = 0 \qquad (1)$$

$$\frac{D\mathbf{v}}{Dt} = -\frac{1}{\rho}\nabla p + \frac{1}{\rho}\nabla \cdot \mathbb{T} + \mathbf{g}, \qquad (2)$$

where ρ is the fluid density, $\mathbf{v}$ is the fluid velocity, p is pressure, $\mathbb{T}$ is the second-order deviatoric stress tensor, $\mathbf{g}$ represents the gravitational acceleration, and ∇ denotes the gradient operator $(\partial/\partial x, \partial/\partial y, \partial/\partial z)$.

3.1. Solving Incompressibility

The prediction-correction scheme [SL03] is a common method to solve incompressibility, which is composed of two steps. In the first step, an explicit Euler forward integration is applied to get the intermediate particle velocity and position. At this time, only viscous force and gravitational force are considered. The intermediate particle velocity and position are obtained as

$$\mathbf{v}^* = \mathbf{v}^t + \left(\frac{1}{\rho}\nabla \cdot \mathbb{T} + \mathbf{g}\right)\Delta t$$

$$\mathbf{x}^* = \mathbf{x}^t + \mathbf{v}^*\Delta t.$$

Here $\mathbf{v}^t$ and $\mathbf{x}^t$ are the particle velocity and position at tth integration step, $\mathbf{v}^*$ and $\mathbf{x}^*$ are the intermediate particle velocity and position, Δt is integration time step.

However, incompressibility is not satisfied at the moment. Thus, a second correction step is required to enforce incompressibility. The pressure term plays an important role in this step to project the intermediate velocity field onto a divergence-free space. The relevant steps are

$$\Delta \mathbf{v}^* = -\frac{1}{\rho^*} \nabla p^{t+1} \Delta t$$

$$\mathbf{v}^{t+1} = \mathbf{v}^* + \Delta \mathbf{v}^*,$$

where $\Delta \mathbf{v}^*$ is the velocity increment during correction step, $\mathbf{v}^{t+1}$ and p^{t+1} represent particle velocity and pressure at $(t + 1)$th integration step, ρ^* represents the intermediate density computed from the intermediate particle position $\mathbf{x}^*$. Then the mass conservation equation can be discretized at intermediate time and result in the following equation

$$\frac{1}{\rho_0} \frac{\rho^* - \rho_0}{\Delta t} + \nabla \cdot \mathbf{v}^* = 0.$$

Here ρ_0 is the reference density. Substitute $\mathbf{v}^* = \mathbf{v}^{t+1} - \Delta \mathbf{v}^*$ and $\Delta \mathbf{v}^*$ into the above equation and assume $\mathbf{v}^{t+1}$ is divergence-free, the pressure Poisson equation is obtained as follows

$$\nabla \cdot \left(\frac{1}{\rho^*} \nabla p^{t+1} \right) = \frac{\rho_0 - \rho^*}{\rho_0 \Delta t^2}. \tag{3}$$

After solving the pressure Poisson equation, the new velocity $\mathbf{v}^{t+1}$ and position $\mathbf{x}^{t+1}$ can be easily updated.

3.2. SPH Formulations

SPH is a fully particle-based technique. In the basic SPH, fluid is represented by a finite number of particles. Each particle carries physical quantities such as mass, velocity, etc, and occupies individual space. A physical variable $f(\mathbf{x})$ can be interpolated by the surrounding particles within the support domain of the smoothing function W

$$f(\mathbf{x}) = \sum_j \frac{m_j}{\rho_j} f_j W(\mathbf{x} - \mathbf{x}_j, h), \tag{4}$$

where m_j, ρ_j and $\mathbf{x}_j$ are material mass, density and position of neighbouring particle j, and W is the weighting kernel with smoothing length h.

Applying this approximation to the forces as defined in [Mon94, MCG03], the following two symmetric forms are obtained

$$\mathbf{F}_i^{\text{pressure}} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W_{ij} \tag{5}$$

$$\mathbf{F}_i^{\text{viscosity}} = \mu \sum_j m_j \frac{\mathbf{v}_j - \mathbf{v}_i}{\rho_j} \nabla^2 W_{ij}. \tag{6}$$

Here μ represents the viscosity coefficient, p_i and $\mathbf{v}_i$ are the pressure and velocity of particle i, W_{ij} is the smoothing function of particle i evaluated at particle j written as $W(\mathbf{x}_i - \mathbf{x}_j, h)$.

The most popular density approximation is the summation approach which well represents the essence of SPH approximation. However, due to the particle deficiency near the boundary, the density summation approach will smooth out the density of the concerned particles, leading to spurious results. A lot of modifications have been proposed to improve the accuracy of this approach. In our implementation, we use the technique defined in [RL96] to improve the accuracy near free surface boundaries and material interfaces.

$$\rho_i = \frac{\sum_j m_j W_{ij}}{\sum_j \left(\frac{m_j}{\rho_j} \right) W_{ij}}.$$

To solve incompressibility, WCSPH introduces artificial compressibility which assumes that every theoretically incompressible fluid is actually compressible. [Mon94] applied the following equation of state to model water's free surface flows:

$$p_i = B \left(\left(\frac{\rho_i}{\rho_0} \right)^\gamma - 1 \right), \tag{7}$$

where $\gamma = 7$ is used in most circumstances and B governs the relative density fluctuation. However, according to the Courant-Friedrichs-Lewy(CFL) condition, to gain a relatively small density error requires sufficiently small time step which increases the total computation time tremendously. To avoid the time step restriction, one can alternatively solve a global pressure Poisson equation as formulated in Equation 3. In the projection method [CR99] , the intermediate velocity field is projected onto a divergence-free space by solving the pressure Poisson equation. Although larger time step can be used, solving the elliptic pressure Poisson equation increases the total work per time step, especially for large systems. Other methods such as MPS method [PTB*03] and Hybrid Particle-Mesh Method [LKO05] also need to solve the global Poisson equation which is time-consuming. However, our method aims to overcome the above problems and explicitly solves the pressure with a local Poisson method, which proves to be accurate and efficient.

4. Local Poisson Solver

We first consider Poisson equation [Eva98]

$$-\Delta u = f \tag{8}$$

in continuous space, here f is a known scalar function and u is the scalar function to be solved. As we know, except for some specific f, we can hardly obtain the analytic solution. Fortunately, we can find the fundamental solution of Laplace's equation $\Delta \Phi = 0$ as follows:

$$\Phi(\mathbf{x}) : = \begin{cases} -\dfrac{1}{2\pi} \log |\mathbf{x}| & (n = 2) \\ \dfrac{1}{4\pi} \dfrac{1}{|\mathbf{x}|} & (n = 3). \end{cases} \tag{9}$$

Here n means dimensionality. Then we could turn the Poisson equation from differential form to the following integral form [Eva98]

$$u(\mathbf{x}) = \int_{\mathbb{R}^n} \Phi(\mathbf{x} - \mathbf{x}') f(\mathbf{x}') d\mathbf{x}'$$

$$= \begin{cases} -\dfrac{1}{2\pi} \displaystyle\int_{\mathbb{R}^2} \log(|\mathbf{x} - \mathbf{x}'|) f(\mathbf{x}') d\mathbf{x}' & (n = 2) \\ \dfrac{1}{4\pi} \displaystyle\int_{\mathbb{R}^3} \dfrac{f(\mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|} d\mathbf{x}' & (n = 3). \end{cases} \tag{10}$$

The constant coefficient of $\Phi(\mathbf{x})$ is chosen to meet the Poisson equation.

In the next section, we will discretize Equation 10 to solve the pressure Poisson equation.

4.1. Pressure Derivation

For the purposes of brevity, we only give pressure derivation in three dimensional space and list the two dimensional form directly at the end of this section.

As discussed in Section 3.1, the key to achieve incompressibility of the fluid is to solve the pressure Poisson equation in Equation 3. A simple transformation can be applied to obtain the following equation

$$-\Delta p^{t+1} = \frac{(\rho^* - \rho_0)\rho^*}{\rho_0 \Delta t^2} - \frac{1}{\rho^*} \nabla \rho^* \cdot \nabla p^{t+1}. \tag{11}$$

As we know, the approximation of the spatial derivative will be smoothed out in SPH. Besides, the time step Δt is small. Thus, the second term of the right hand side of Equation 11 will be much smaller than the first term. We can then eliminate the second term and get

$$-\Delta p^{t+1} = \frac{(\rho^* - \rho_0)\rho^*}{\rho_0 \Delta t^2}, \tag{12}$$

which is a standard form of the Poisson equation in Equation 8, and no unknown variables exist in the right hand side of Equation 12 anymore. Invoking the Equation 10, we obtain

$$p^{t+1}(\mathbf{x}) = \frac{1}{4\pi \rho_0 \Delta t^2} \int_{\Omega(\mathbf{x})} \frac{(\rho^*(\mathbf{x}') - \rho_0) \rho^*(\mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|} d\mathbf{x}' \tag{13}$$

where $\Omega(\mathbf{x})$ represents the 'local pressure integration domain' for position $\mathbf{x}$ as it varies from place to place in our implementation. More details can be found in Section 4.3.

In the SPH method, since the entire system is represented by a finite number of particles, the continuous integral representation for equation 13 should be written in the form of discretized particle approximation as follows

$$p_i^{t+1} \cong \frac{1}{4\pi \rho_0 \Delta t^2} \sum_j \left(\int_{\Omega_{i|j}} \frac{(\rho^*(\mathbf{x}') - \rho_0) \rho^*(\mathbf{x}')}{|\mathbf{x}_i - \mathbf{x}'|} d\mathbf{x}' \right) \tag{14}$$

Here, we use the SPH convection and mark the pressure of particle i at the position of x_i as p_i^{t+1}. $\Omega_{i|j}$ represents the volume occupied by particle j with a spherical shape centred at $\mathbf{x}_j$ in the local pressure integration domain Ω_i, which means $\bigcup_j \Omega_{i|j} \cong \Omega(x_i)$. From Equation 14, we can find that the particle pressure is contributed by its neighbouring particles in the local pressure integration domain.

If we make the simplistic assumption that $\rho^*(x')$ is constant throughout $\Omega_{i|j}$ for each particle, Equation 14 can be converted to

$$p_i^{t+1} \cong \sum_j \left(\frac{(\rho_j^* - \rho_0)\rho_j^*}{4\pi \rho_0 \Delta t^2} \int_{\Omega_{i|j}} \frac{1}{|\mathbf{x}_i - \mathbf{x}'|} d\mathbf{x}' \right) \tag{15}$$

where ρ_j^* is the intermediate density for particle j.

To calculate the integral in Equation 15, we divide the domain Ω_i into the following two separated regions

$$\Omega_{i|\text{near}} = \{\mathbf{x} \,|\, |\mathbf{x} - \mathbf{x}_i| \leq \varepsilon\} \cap \Omega_i$$
$$\Omega_{i|\text{far}} = \{\mathbf{x} \,|\, |\mathbf{x} - \mathbf{x}_i| > \varepsilon\} \cap \Omega_i$$

where ε is a small value which will be discussed later in this section. Intuitively, $\Omega_{i|\text{near}}$ represents the nearby area of particle i where the integrand $1/|\mathbf{x}_i - \mathbf{x}'|$ changes rapidly, $\Omega_{i|\text{far}}$ represents the area where the integrand $1/|\mathbf{x}_i - \mathbf{x}'|$ changes relatively slow.

Now we will derive the approximate integral method for particles in $\Omega_{i|\text{near}}$ and $\Omega_{i|\text{far}}$ (hereinafter referred to as near particles and far particles), respectively. For far particles, the integrand can be approximated by a constant function sampled at the particle centre $\mathbf{x}_j$, thus we can get

$$p_{i|\text{far}}^{t+1} = \sum_{\mathbf{x}_j \in \Omega_{i|\text{far}}} \frac{(\rho_j^* - \rho_0)\rho_j^*}{4\pi \rho_0 \Delta t^2} \int_{\Omega_{i|j}} \frac{1}{|\mathbf{x}_i - \mathbf{x}'|} dx'$$

$$\cong \sum_{|\mathbf{x}_{ij}| > \varepsilon} \frac{(\rho_j^* - \rho_0)\rho_j^*}{4\pi \rho_0 \Delta t^2} \frac{1}{|\mathbf{x}_i - \mathbf{x}_j|} V_j^* \tag{16}$$

$$= \sum_{|\mathbf{x}_{ij}| > \varepsilon} m_j \frac{(\rho_j^* - \rho_0)}{4\pi \rho_0 \Delta t^2 |\mathbf{x}_i - \mathbf{x}_j|}$$

where $p_{i|\text{far}}^{t+1}$ is the pressure integral over $\Omega_{i|\text{far}}$, V_j^* is the volume of particle j equal to m_j/ρ_j^* and $\mathbf{x}_{ij}$ is short for
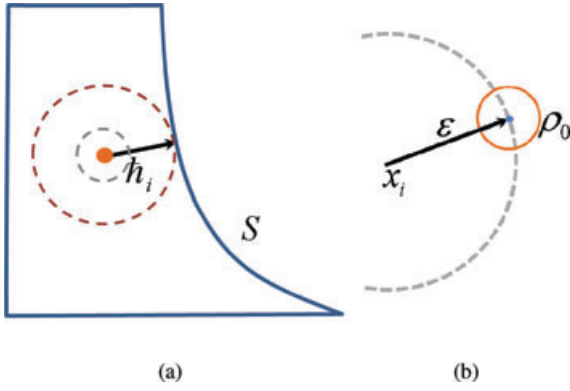
Figure 1: *(a) Local pressure integration domain for particle i. (b) A reference particle resides at the separating interface.*

Figure 2: *Simulation results of a 2D dam-break case with different methods. (a) WCSPH. (b) LPSPH with global pressure integration domain. (c) LPSPH with local pressure integration domain.*

$\mathbf{x}_i - \mathbf{x}_j$. However, no constant function is adequate for near particles, thus direct integral method is applied to solve this problem. Before this, we first move particle j to get a spherical symmetry integral domain about x_i, indicating $\mathbf{x}_i = \mathbf{x}_j$ during integration process, then the integral can be done in spherical coordinates as follows

$$
\begin{aligned}
p_{i|\mathrm{near}}^{t+1} &= \sum_{\mathbf{x}_j \in \Omega_{i|\mathrm{near}}} \frac{(\rho_j^* - \rho_0)\rho_j^*}{4\pi \rho_0 \Delta t^2} \int_{\Omega_{i|j}} \frac{1}{|\mathbf{x}_i - \mathbf{x}'|} d\mathbf{x}' \\
&\cong \sum_{|\mathbf{x}_{ij}| \le \varepsilon} \frac{(\rho_j^* - \rho_0)\rho_j^*}{4\pi \rho_0 \Delta t^2} \int_{\Omega_{i|j}} \frac{1}{|\mathbf{x}_j - \mathbf{x}'|} d\mathbf{x}' \quad (17) \\
&= \sum_{|\mathbf{x}_{ij}| \le \varepsilon} \frac{(\rho_j^* - \rho_0)\rho_j^* r_j^2}{2\rho_0 \Delta t^2}
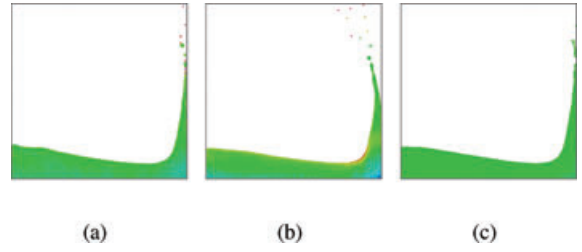\end{aligned}
$$

where $p_{i|\mathrm{near}}^{t+1}$ is the pressure integral over $\Omega_{i|\mathrm{near}}$ and r_j represents the radius of particle j. Now we can get pressure by

$$
p_i^{t+1} = p_{i|\mathrm{near}}^{t+1} + p_{i|\mathrm{far}}^{t+1}.
$$

To compute ε, we consider a particle with reference density ρ_0 located in a position shown in Figure 1 (b). For this particle, it should be equivalent to use either of the approximate integral method in Equation 16 or 17, that is

$$
\frac{1}{|\mathbf{x}_i - \mathbf{x}_j|} V_j^* = \int_{\Omega_{i|j}} \frac{1}{|\mathbf{x}_j - \mathbf{x}'|} d\mathbf{x}'
$$

with the constant part $\frac{(\rho_j^* - \rho_0)\rho_j^*}{4\pi \rho_0 \Delta t^2}$ eliminated at both sides. By denoting $\varepsilon = |\mathbf{x}_i - \mathbf{x}_j|$, we can easily obtain $\varepsilon = \frac{2}{3}r_0$, here r_0 represents the radius of the reference particle.

Similarly, the two dimensional formulations are listed below without derivation

$$
p_{i|\mathrm{near}}^{t+1} = \sum_{|\mathbf{x}_{ij}| \le \varepsilon} \frac{(2\log(r_j) - 1)(\rho_j^* - \rho_0)\rho_j^* r_j^2}{2\rho_0 \Delta t^2}
$$

$$
p_{i|\mathrm{far}}^{t+1} = \sum_{|\mathbf{x}_{ij}| > \varepsilon} m_j \frac{(\rho_j^* - \rho_0)\rho_j^* \log|\mathbf{x}_j - \mathbf{x}_i|}{\rho_0 \Delta t^2}
$$

4.2. Local Pressure Integration Domain

As indicated by Equation 10, we can simply define the pressure integration domain Ω_i as the whole problem domain, and we define it as global. Otherwise, it will be local. However, there are two main reasons for us not applying the global strategy: First, the fluid state changes all the time during the dynamic fluid simulation, the particles in separated areas should have no effects on each other. Secondly, if we do not use any other accelerating algorithm such as the fast multipole method [Gre88], a computational complexity of $O(N^2)$ is needed to compute the pressure for all particles. It is usually a disaster if we want to get a realistic fluid animation. To address these two problems and meet the core idea of SPH, we adopt to use a geometric local domain (red circle in Figure 1(a)) to solve the Poisson problem and define Ω_i as follows

$$
\Omega_i = \{\mathbf{x} |\ |\mathbf{x} - \mathbf{x}_i| \le \hbar_i, \hbar_i = \min_{\mathbf{y} \in S} |\mathbf{y} - \mathbf{x}_i|\}
$$

where S represents the fluid boundary, $\hbar_i$ measures the shortest distance from particle i to S which can be effectively computed using the redistancing method defined in [APKG07]. From the definition, we find $\hbar_i$ decreases as the particle approaches fluid boundary, thus less computational effort is needed for computing particle pressure near the boundary than in the interior. With this strategy, Figure 2 gives the simulation results of a testing 2D dam-break case. We can find our local method (Figure 2c) makes the fluid shape well agree with the WCSPH result(Figure 2a) while large

deviations can be found in Figure 2(b) where we use the whole problem domain for pressure computing.

However, in large-scale fluid simulation, the particle number in Ω_i could be still very large for interior particles. We thus bring in a const $\hbar_{\max} \in [0, \infty)$ to balance the accuracy and efficiency

$$\hbar_i = \min(\hbar_i, \hbar_{\max})$$

Here two special cases should be noted: As ∞ is the supremum of $sup\{\hbar_i\}$ for all simulations, here we refer to $\hbar_{\max} = \infty$ as the simulation case with $\hbar_i$ unrestricted. In the case of $\hbar_{\max} = 0$, for p_i, we only need to compute the pressure contribution by the particle itself, thus the pressure solver in Equation 17 can be reduced to the following simple form in three dimension

$$p_i = \frac{(\rho_i^* - \rho_0)\rho_i^* r_i^2}{2\rho_0 \Delta t^2} \qquad (n = 3)$$

which has a great similarity with the EOS in the form of Equation 7. Besides, the two dimensional form for $\hbar_{\max} = 0$ is as follows

$$p_i = \frac{(2\log(r_i) - 1)(\rho_i^* - \rho_0)\rho_i^* r_i^2}{2\rho_0 \Delta t^2} \qquad (n = 2)$$

If we designate $\rho_{err_i}^* = \rho_i^* - \rho_0$ and take a further comparison with the pressure form given by PCISPH, we can find that our pressure form is quadratic about $\rho_{err_i}^*$ while the form given by PCISPH is linear.

4.3. Prediction-Correction Framework

Although our LPSPH method shows a higher density error decay rate over WCSPH as discussed in Section 5.1, we have to control the density error under a user-defined threshold η strictly. To achieve this goal, we integrate our method into the prediction-correction framework defined in [SP09] and call it *iterative*-LPSPH. The modified pseudocode is shown in Algorithm *i*-LPSPH.

As we can see in the pseudocode, the particle velocities and positions are temporarily forwarded (line 7~8) without considering pressure forces. The resulting intermediate densities (line 14) are then used to estimate the required pressure with our local Poisson method (line 15) to update the intermediate velocities and positions (line 18~19) until the particle density fluctuation is smaller than the predefined threshold η (line 21). Here $N_i(t)$ represents the neighbouring particles in the support domain and $\aleph_i(t)$ represents the particles in the local pressure integral domain Ω_i for each particle i. Compared to WCSPH, our method usually needs to compute extra pressure neighborhoods $\aleph_i(t)$ except for a particular $\hbar_{\max}$, which will be further discussed in Section 5. However, in our real implementation we only recompute pressure radius $\hbar_i$ (line 3) every 20 steps if necessary and no

significant visual differences can be found. From the pseudocode, we note that the *non-iterative* LPSPH is just a special case of *iterative*-LPSPH with η sufficiently large. It also indicates that the shift between the *non-iterative* LPSPH and *iterative*-LPSPH is easy.

Algorithm *i*-LPSPH

|	
1	**while** animating **do**
2	**for each** particle i
3	compute $\hbar_i$ as in 4.3
4	find neighbors $N_i(t)$
5	compute forces $\mathbf{F}_i^{\nu,\mathbf{g}}(t)$
6	**for each** particle i
7	predict velocity $\mathbf{v}_i^* = \mathbf{v}_i(t) + \Delta t * \mathbf{F}_i^{\nu,\mathbf{g}}(t)/m_i$
8	predict position $\mathbf{x}_i^* = \mathbf{x}_i(t) + \Delta t * \mathbf{v}_i^*$
9	**do**
10	**for each** particle i
11	find neighbors $N_i(t)$
12	find pressure neighborhoods $\aleph_i(t)$
13	**for each** particle i
14	predict density ρ_i^*
15	compute p_i as in 4.2
16	compute pressure force $\mathbf{F}_i^p(t)$
18	update velocity $\mathbf{v}_i^* += \Delta t * \mathbf{F}_i^p(t)/m_i$
19	update position $\mathbf{x}_i^* += \Delta t^2 * \mathbf{F}_i^p(t)/m_i$
20	compute density variation ρ_{err}^*
21	**while** ($\rho_{err}^* > \eta$)
22	**for each** particle i
23	update velocity $\mathbf{v}_i(t+1) = \mathbf{v}_i^*$
24	update position $\mathbf{x}_i(t+1) = \mathbf{x}_i^*$

---

5. Results and discussion

To assess the accuracy and efficiency of the LPSPH method, testing cases including both 2D and 3D have been implemented on a PC with an Intel Dual-Core 2.8 GHz CPU and 3GB RAM. In all animations, the surface of the fluid is defined with the approach presented in [APKG07] and rendered using POV-Ray (http://www.povray.org) after extracting the fluid surface using a marching cubes algorithm [LC87].

5.1. Density Error Analysis

We first set up a 2D dam break case with 4639 particles to compare the density errors of both the commonly used WCSPH and our new LPSPH method with $\hbar$ unrestricted. As we have found in WCSPH, after temporally integrating the fluid forward, large density variations could be caused by two major reasons: solid wall boundary treatment and free surface topological change. The former usually causes fluid to be compressed, and the resulted error will then spread out radially as depicted in Figure 3(a) and decay as time
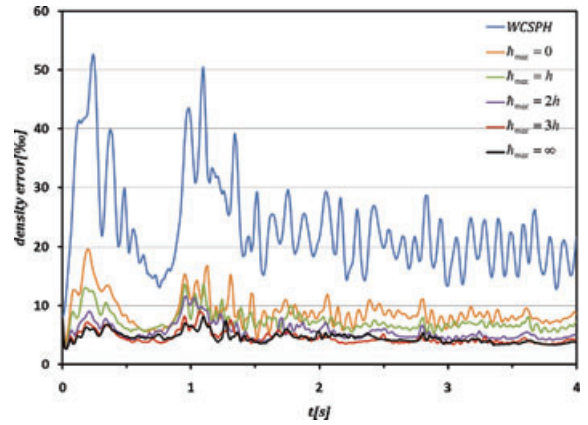
(a) WCSPH   **(b) LPSPH**

(c) WCSPH   **(d) LPSPH**
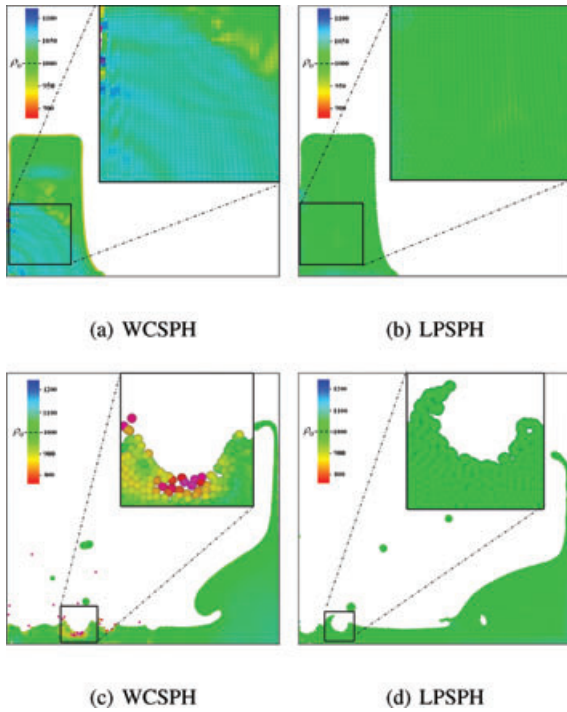
Figure 3: *Comparison of density error pattern caused by boundary treatment (up) and topological change (down).*

Figure 4: *Comparison of density errors over time between our local Poisson solver with different maximum pressure radius and EOS-based solver.*
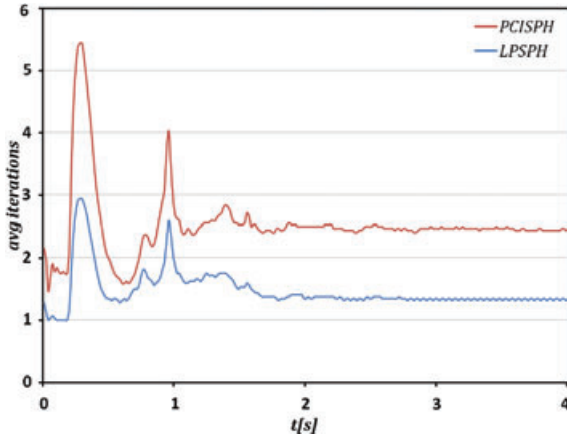
Table 1: *Statistics of different simulations. Notice how the mean value decreases and computation time increases as $\hbar_{max}$ increases. Also notice the differences between the simulations with and without redistancing when $\hbar_{max} = h$*

Method	Mean	Std.	Redistance	t_{sim}[min]
WCSPH	23.28	8.34	−	6.36
$\hbar_{max} = 0$	8.96	2.80	−	8.26
$\hbar_{max} = h$	7.44	2.32	−	10.96
$\hbar_{max} = h$	6.84	1.85	$\checkmark$	12.44
$\hbar_{max} = 2h$	5.63	1.62	$\checkmark$	15.29
$\hbar_{max} = 3h$	4.57	1.19	$\checkmark$	19.32
$\hbar_{max} = \infty$	4.58	1.03	$\checkmark$	29.64

evolves. The spread speed and decay rates are closely related to the smooth radius h, the time step size Δt and the stiffness parameter k of EOS equation. It's always a cumbersome work to choose the appropriate parameter value for the EOS equation. However our method effectively overcomes this difficulty and shows higher density error decay rates.

Here we choose a relatively large time step size $\Delta t = 5 \times 10^{-4}$s and set the reference density $\rho_0 = 1000\,\text{kg/m}^3$. As we can see in Figure 3(b), the interior density errors have been greatly eliminated at $t = 0.1s$ while we can still find obvious over estimated density errors in the interior part of Figure 3(a). On the other hand, under estimated density errors usually occur when fluid topology changes as depicted in Figure 3(c) where a small drop of water splashes into a larger body of water. A similar scene can also be noted in Figure 3(d) with LPSPH method in which we do not find any significant errors. We refer the readers to the accompanying video to assess the differences in dynamic simulations.

As the choice of $\hbar_{max}$ can significantly affect the results, more detailed numerical analysis is needed. Along with the WCSPH method, we run different simulations of LPSPH with varying $\hbar_{max}$ and plot the density errors over time in Figure 4 . The results for $\hbar_{max} = 0$, $\hbar_{max} = h$, $\hbar_{max} = 2h$ and $\hbar_{max} = 3h$ show a overall decreasing density error at the cost of increasing computing expenses as shown in Table 1. The case with $\hbar$ unrestricted corresponding to $\hbar_{max} = \infty$ pro-

vides the best result of accuracy and stability as the mean and standard deviation of the density error indicate, but requires the most expensive computational cost. Thus, balance between accuracy and efficiency is needed. In this 2D case, since no more significant accuracy could be further obtained when $\hbar_{max}$ exceeds $3h$, we suggest to choose $\hbar_{max} \in [0, 3h]$ according to the different requirement. In addition, when we set $\hbar_{max} = h$, both the simulations with and without redistancing are executed. Although minor advantage can be noted in Table 1 with redistancing, we can always improve efficiency by avoiding redistancing.

5.2. Convergence Analysis

In this section, we integrate our pressure solver into the predictive-corrective framework and perform a convergence analysis for the 3D water drop case, the predefined maximally allowed density fluctuation with respect to the reference density is $\eta = 1\%$.

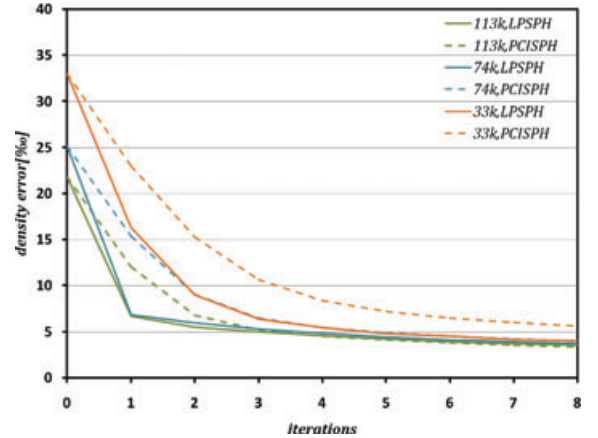**Figure 5:** *Comparison of iteration numbers over time between PCISPH and iterative-LPSPH.*

Figure 6: *Several convergence examples with different resolutions at $t = 0$. Notice how fast the density error reaches the predefined threshold $\eta = 1\%$ with both methods.*

For efficiency, we here set $\hbar_{max} = 0$ to make each iteration as simple as possible. Thus, no extra effort is needed to do redistancing and search neighboring particles. According to [SP09], a minimum of three iterations is required to achieve a low level of pressure fluctuations. However, in our implementation, we don't apply this strategy for both PCISPH and our LPSPH method. The iteration number is plotted over time in Figure 5. It can be seen that the iteration number of LPSPH is approximately halved compared to PCISPH. After 4s of simulation, the former gets an average iteration of 1.47 while the later 2.53. As we know, at least 1 iteration is required, thus we only take 0.47 extra iteration per physics step by using our method in this example, which is less than a third of that needed for PCISPH. Figure 6 shows several examples (33k, 74k and 113k) of the convergence within a single physics update step where we can also find faster convergence rates with our method.

5.3. Performance Comparison and Visual Results

In this section, both the previously described 2D and 3D simulations are used for a comparison between WCSPH, PCISPH and our iterative-LPSPH method.

In WCSPH case, the time step is limited by CFL condition, the viscous diffusion term and the force terms [Mon92]:

$$\Delta t = \min\left(0.25\min_{i}\left(\frac{h}{|\mathbf{f}_i|}\right), 0.4\left(\frac{h}{c_s(1+0.6\alpha)}\right)\right)$$
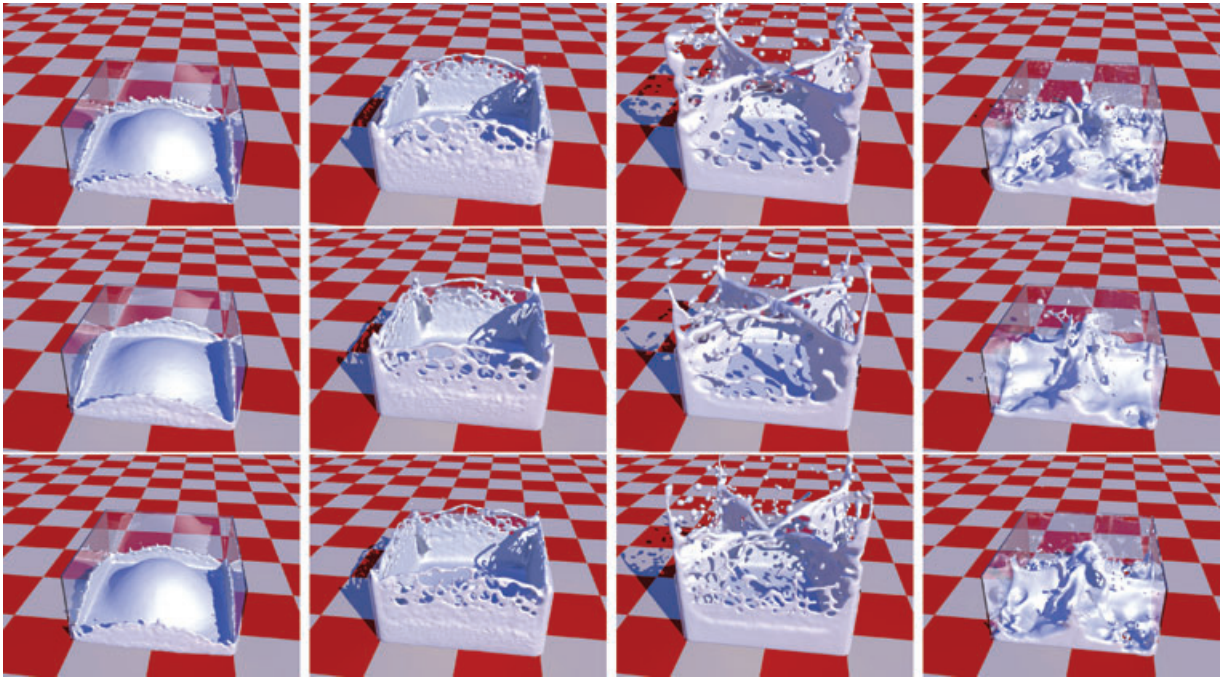
with $\mathbf{f}_i$ denoting external forces, α denoting the viscous constant 0.5 according to [BT07] and c_s denoting the speed of sound in the fluid. To meet the maximally allowed density error $\eta = 1\%$ from reference density, an available method to compute the pressure constant B is given according to the following equation [BT07]:

$$B = \frac{\rho_0 c_s^2}{\gamma}, c_s = \sqrt{\frac{2gH}{\eta}},$$

where H represents the height of the scene. We strongly refer the reader to the original paper to get more details. However, more tests are usually needed to adjust B to get the appropriate value.

In our implementation for PCISPH, we use a precomputed single scaling factor δ which is evaluated for a prototype particle with a filled neighbourhood. The performance

Table 2: *Comparison between WCSPH, PCISPH and LPSPH.*

Case	Method	#p	B	$\Delta t[s]$	Δt ratio	$t_{sim}[min]$	Avg Iterations	Speed-up
2D Dam-Break	WCSPH	4639	1.81×10^5	2.92×10^{-5}	–	101.50	–	–
2D Dam-Break	PCISPH	4639	–	1.00×10^{-3}	34.25	9.77	3.35	10.39
2D Dam-Break	i-LPSPH	4639	–	1.00×10^{-3}	34.25	6.19	1.70	16.38
3D Water Drop	WCSPH	113k	2.22×10^5	1.95×10^{-5}	–	6191.83	–	–
3D Water Drop	PCISPH	113k	–	1.00×10^{-3}	51.28	369.18	2.53	16.78
3D Water Drop	i-LPSPH	113k	–	1.00×10^{-3}	51.28	272.05	1.47	22.76

Figure 7: *The similarity of visual results simulated by both WCSPH (top panel), PCISPH (middle panel) and LPSPH (bottom panel) with 113k particles.*
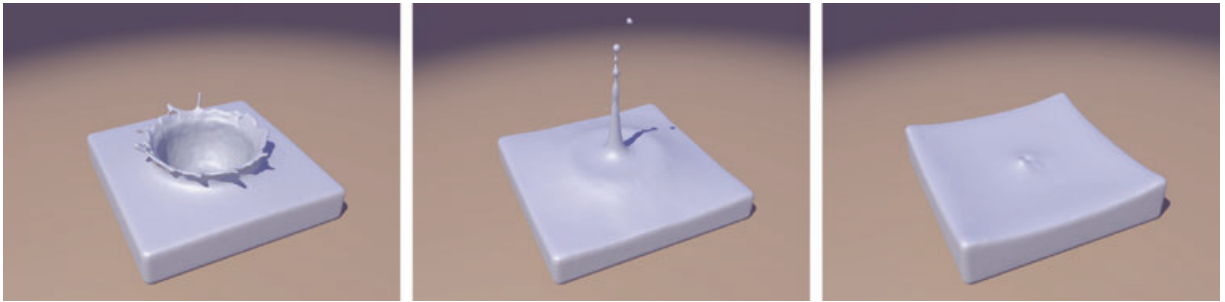
Figure 8: *An animation of a water drop splashing into a larger body of water.*
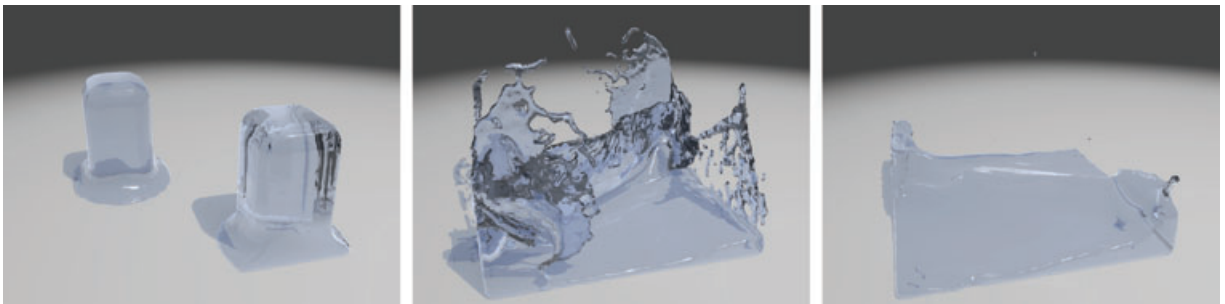
Figure 9: *A double dam break with 86k particles.*

measurements and simulation data for the 2D and 3D cases are summarised in Table 2. The physical behaviour and visual results are compared in Figure 7 with the 3D case of 113k particles where we can see full agreement of the three methods with only very minor detail differences. Besides, more examples with varying particle resolutions are executed. Figure 8 shows a double dam break with 86k particles which throws up a thin sheet of water that runs diagonally across the tank. Figure 9 presents an animation of a water drop that splashes into a larger body of water, causing a realistic water crown and small bouncing droplets. The total particle number is 188k here. In all examples, a η of 1% is enforced to eliminate compression artifacts and enhance visual effects.

6. Conclusion and Future Work

We present a local Poisson SPH Method to solve viscous incompressible fluid and preserve both the high computing efficiency per physical step of WCSPH and the large time step of ISPH. Unlike previous pressure solvers, we solve the pressure Poisson equation with a new integral method. First, the differential pressure Poisson equation is converted into an integral form. Secondly, a discretization technology is used to convert the continuous integral form to a discretized summation over all the particles in the local pressure integration domain. Finally, we integrate our pressure solver method into the predictive-corrective framework, achieving a flexible density error control. Experiments show the advantages of LPSPH both in density errors and convergence rates.

As our method is related to local geometry, we will pay attentions to adaptive sampling as for example, proposed in [APKG07] to break through the $\hbar_{max}$ limitation, thus maintain a relatively constant number of particles in the local pressure integration domain, and it will be a great performance improvement.

References

[APKG07] ADAMS B., PAULY M., KEISER R., GUIBAS L. J.: *Adaptively sampled particle fluids. ACM Transactions on Graphics 26*, 3 (2007), 48–54.

[BLS11] BODIN K., LACOURSIERE C., SERVIN M.: Constraint fluids. *IEEE Transactions on Visualization and Computer Graphics*, 99 (2011), 1–1.

[BT07] BECKER M., TESCHNER M.: Weakly compressible SPH for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (San Diego, USA, 2007), Eurographics Association, pp. 209–217.

[BZZW09] BAO K., ZHANG H., ZHENG L., WU E.: Pressure corrected SPH for fluid animation. *Computer Animation and Virtual Worlds 20* 2-3, (2009), 311–320.

[CFL67] COURANT R., FRIEDRICHS K., LEWY H.: On the partial difference equations of mathematical physics. *IBM Journal of Research and Development 11*, 2 (1967), 215–234.

[CR99] CUMMINS S., RUDMAN M.: An SPH projection method. *Journal of computational physics 152*, 2 (1999), 584–607.

[DG96] DESBRUN M., GASCUEL M.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Proceedings of the Computer Animation and Simulation* (1996), vol. 96, Citeseer, Poitiers, France, pp. 61–76.

[Eva98] EVANS L.: *Partial Differential Equations*. American Mathematical Society, 1998.

[GM77] GINGOLD R., MONAGHAN J.: Smoothed particle hydrodynamics-theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society 181* (1977), 375–389.

[Gre88] GREENGARD L.: *The Rapid Evaluation of Potential Fields in Particle Systems*, vol. 1987. the MIT Press, Cambridge, MA, USA, 1988.

[HA07] HU X., ADAMS N.: An incompressible multi-phase SPH method. *Journal of Computational Physics 227*, 1 (2007), 264–278.

[IABT11] IHMSEN M., AKINCI N., BECKER M., TESCHNER M.: A parallel SPH implementation on multi-core CPUS. In *Computer Graphics Forum 30*, 1 (2011), 99–112.

[KAG*05] KEISER R., ADAMS B., GASSER D., BAZZI P., DUTRÉ P., GROSS M.: A unified lagrangian approach to solid-fluid animation. In *Proceedings of the Eurographics Symposium on Point-Based Graphics* (2005), Citeseer, New York, NY, USA, pp. 125–134.

[LAD08] LENAERTS T., ADAMS B., DUTRÉ P.: Porous flow in particle-based fluid simulations. *ACM Transactions on Graphics (TOG) 27*, 3 (2008), 1–8.

[LC87] LORENSEN W., CLINE H.: Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics 21*, 4 (1987), 163–169.

[LD09] LENAERTS T., DUTRÉ P.: Mixing fluids and granular materials. *Computer Graphics Forum 28*, 2 (2009), 213–218.

[LKO05] LIU J., KOSHIZUKA S., OKA Y.: A hybrid particle-mesh method for viscous, incompressible, multiphase flows. *Journal of Computational Physics 202*, 1 (2005), 65–93.

[LTKF08] LOSASSO F., TALTON J., KWATRA N., FEDKIW R.: Two-way coupled SPH and particle level set fluid simulation. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (2008), 797–804.

[Luc77] LUCY L. B.: A numerical approach to the testing of the fission hypothesis. *Astronomical Journal 82*, 3 (1977), 1013–1024.

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *Eurographics/SIGGRAPH Symposium on Computer Animation*. D. Breen and M. Lin (Eds.). San Diego, California (2003), Eurographics Association, pp. 154–159.

[MFZ97] MORRIS J., FOX P., ZHU Y.: Modeling low Reynolds number incompressible flows using SPH. *Journal of Computational Physics 136*, 1 (1997), 214–226.

[Mon92] MONAGHAN J.: Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics 30* (1992), 543–574.

[Mon94] MONAGHAN J.: Simulating free surface flows with SPH. *Journal of computational physics 110* (1994), 399.

[MSKG05] MÜLLER M., SOLENTHALER B., KEISER R., GROSS M.: Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Los Angeles, CA, USA, 2005), ACM, p. 244.

[PTB*03] PREMŽOE S., TASDIZEN T., BIGLER J., LEFOHN A., WHITAKER R.: Particle-Based Simulation of Fluids. In *Computer Graphics Forum 22*, 3 (2003), 401–410.

[RL96] RANDLES P., LIBERSKY L.: Smoothed particle hydrodynamics: some recent improvements and applications. *Computer Methods in Applied Mechanics and Engineering 139*, 1-4 (1996), 375–408.

[RWT11] RAVEENDRAN K., WOJTAN C., TURK G.: Hybrid smoothed particle hydrodynamics. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2011), ACM, Vancouver, Canada, pp. 33–42.

[SBH09] SIN F., BARGTEIL A., HODGINS J.: A point-based method for animating incompressible flow. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2009), ACM, New Orleans, USA, pp. 247–255.

[SL03] SHAO S., LO E.: Incompressible SPH method for simulating newtonian and non-newtonian flows with a free surface. *Advances in Water Resources 26*, 7 (2003), 787–800.

[SP09] SOLENTHALER B., PAJAROLA R.: Predictive-corrective incompressible SPH. In *ACM SIGGRAPH 2009 papers 28*, 3 (2009), 1–6 ACM.

[SSP07] SOLENTHALER B., SCHLÄFLI J., PAJAROLA R.: A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds 18*, 1 (2007), 69–82.