Fast Simulation of Viscous Fluids with Elasticity and Thermal Conductivity Using Position-Based Dynamics

Tetsuya Takahashi[a,b], Tomoyuki Nishita[b,c], Issei Fujishiro[a]

[a] *Keio University*
[b] *UEI Research*
[c] *Hiroshima Shudo University*

Abstract

Viscous fluids are ubiquitous, and reproducing their damped motions has been in demand for many applications. The most prevalent approach to simulating viscous fluids is based on the Navier-Stokes equations and necessitates viscosity integration. However, to simulate viscous fluids in a numerically stable manner, using explicit viscosity integration severely restricts time steps and requires an excessively long period for computation. In this paper, we propose a novel particle-based Lagrangian method for efficiently simulating viscous fluids by adopting position-based constraints. Our method uses the geometric configuration of particles for the positional constraints to approximate the dynamics of viscous fluids using position-based dynamics; thus the method can plausibly generate their motions while allowing for the use of much larger time steps than those previously adopted in the viscous fluid simulations. We also propose an associated boundary-handling scheme for position-based fluids to precisely specify boundary conditions for the constraints. Additionally, we reproduce elastic deformations of materials by controlling the constraints and incorporate thermal conduction into our framework to simulate resultant changes in particle properties and phase transition in the materials. By adjusting parameters, our method can encompass complex motions of fluids with different properties in a unified framework. Several examples demonstrate the effectiveness as well as versatility of our method.

Keywords: Fluid simulation, viscous fluid, position-based dynamics, geometric constraint, elasticity, thermal conductivity.

1. Introduction

Viscous fluids are often found in our daily lives, and their complicated behavior has fascinated us. Unlike inviscid fluids, viscous ones, such as honey, melted chocolate, wax, paints, sap, slime, lava, and blood, exhibit characteristic and interesting motions. Since their damped motions produce visually attractive phenomena and play important roles in enhancing visual reality, these viscous fluids have frequently been adopted in movies and video games.

The most prevalent approach to simulating viscous fluids is based on the Navier-Stokes equations and has successfully simulated viscous fluids [1, 2, 3, 4, 5]. In computer graphics, the equations are usually solved with three different approaches: Eulerian, Lagrangian, or Eulerian-Lagrangian hybrid. We herein focus on the particle-based Lagrangian approach due to its versatility.

To solve the Navier-Stokes equations with particle-based methods, such as *Smoothed Particle Hydrodynamics* (SPH) [6], explicit viscosity integration is often used due to its simplicity and effectiveness. However, this explicit scheme is likely to fail in plausibly reproducing highly viscous fluids under high spatial resolution because, to describe viscous effects based on diffusion equations, we need to satisfy the numerical stability condition $\mu\Delta t/(\rho\Delta x^2) \leq 1/2$ (μ: dynamic viscosity, Δt: time step, ρ: fluid density, and Δx: interparticle distance), according to von Neumann stability analysis [7]. This condition is a dominant factor for determining time steps in the viscous fluid simulations; as a result, $\Delta t \leq O(\rho\Delta x^2/\mu)$ is necessary at the significant sacrifice of computational efficiency.

In this paper, therefore, we propose a novel particle-based Lagrangian method for simulating viscous fluids faster by allowing for the use of larger time steps with position-based constraints. The core idea of our method is to take full advantage of the geometric configuration of particles for the positional constraints to approximate the dynamics of viscous fluids. Since our method does not require solving the diffusion equations, we can ignore the restriction $\Delta t \leq O(\rho\Delta x^2/\mu)$; thus, time steps with our method can be much larger than the limited time steps for explicit viscosity integration.

We use position-based fluids of Macklin and Müller [8] as our fluid solver, where fluid is simulated using position-based dynamics [9, 10], which directly corrects particle positions to produce plausible motions of objects. To precisely specify boundary conditions with the positional constraints, we also propose an associated boundary-handling scheme for position-based fluids with non-fluid particles on object surfaces. Although the position-based method is less accurate than the traditional force-based one, our method can generate visually plausible viscous fluid motions. Figure 1 illustrates the effect of our method with an example of melted chocolate.

As an extension to [11], we newly added handling of elastic materials and thermal conduction to our framework and improved the constraint for viscosity to make it easier to produce

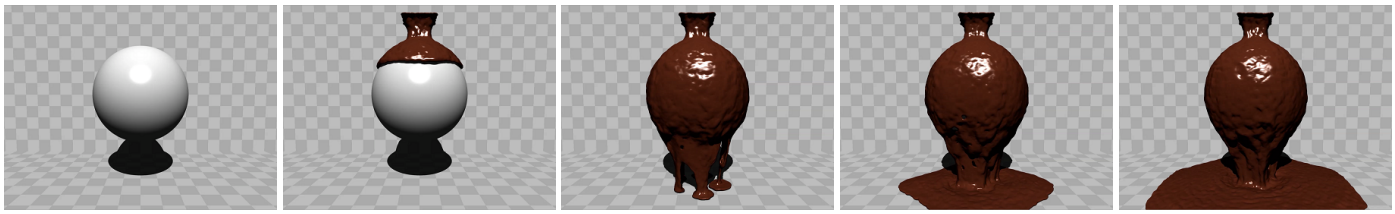


Figure 1: Melted chocolate running down the surface of a white ball. With our method, chocolate flows on and sticks to the ball, and slowly flows over the floor, due to the geometric constraints between particles, which approximate the dynamics of viscous fluids by restricting movements of the particles.

desirable fluid motions by reducing the number of parameters.

In summary, our key contributions are three-fold:

- Much larger time steps are available to simulate viscous fluids than those previously used to numerically stabilize the simulation with explicit viscosity integration by avoiding the strict time step restriction $\Delta t \leq O(\rho \Delta x^2/\mu)$;
- The associated boundary-handling scheme for position-based fluids properly addresses the position-based constraints on object surfaces; and
- Various behaviors of fluids with different properties due to thermal conduction can be generated and controlled by adjusting parameters in a unified framework.

2. Related Work

We briefly explain methods for simulating viscous fluids and techniques for particle-based methods.

Viscous fluids As a versatile framework for various types of materials, the Lagrangian finite element method has been proposed and accurately simulated the dynamics of viscous fluids [12, 13, 14, 15]. However, this method is inappropriate for fast simulation due to the high computational cost of remeshing and solving linear systems.

Bergou et al. [16] and Batty et al. [17] proposed a discrete model specialized for describing dimensionally reduced materials and successfully simulated the distinctive behavior of viscous threads and sheets, respectively, yet their methods are inapplicable to three-dimensional viscous volumes.

Desbrun and Gascuel [18] adopted SPH and simulated viscous materials with an artificial viscosity term [19]. This term was also used by Stora et al. [20] for lava simulation. Müller et al. [6] proposed a viscosity term denoted by the Laplacian operator and simulated dynamic fluid with viscosity. Solenthaler et al. [21] combined this term with an elastic force term to simulate fluid with viscosity and elasticity and simulated changing particle properties and phase transitions in materials by solving heat equations in a unified framework. Becker et al. [22] improved Solenthaler et al.'s method [21] to handle rotational motions of elastic objects. Paiva et al. [5] used a generalized Newtonian model for viscous fluid simulation and adopted XSPH [23], which smoothes velocities for coherent particle movements. For simulating viscoelastic materials, Mao and Yang [24] introduced a new elastic force term, which is a nonlinear corotational Maxwell model, into the Navier-Stokes equations. Chang et al. [25] used a simplified Maxwell model, which drops a rotational tensor, and simulated viscoelastic materials including the effect of thermal conduction. A Maxwell model was also used to simulate the coiling phenomenon [26].

Recently, methods that combine SPH with another technique have been proposed. Gerszewski et al. [27] presented a method that uses an affine transform to approximate the motions of neighboring particles for reproducing elastoplastic materials. Takamatsu and Kanai [28] simulated viscoelastic materials by combining SPH with shape matching. Dagenais et al. [29] reproduced viscous fluid motions in SPH by adding extra forces that move particles to their original positions.

Several intuitive methods have also been proposed for viscous materials. Miller and Pearce [30] and Terzopoulos et al. [31] reproduced viscous materials using a spring-based model that computes the repulsion and attraction forces between particles. This idea was also adopted by Steele et al. [32] and Tamura et al. [33]. Clavet et al. [34] extended this model to simulate materials that exhibit elasticity, viscosity, and plasticity.

Since particles are controlled on the basis of their geometric relations without estimating force fields, our method can be categorized as a spring-based method. However, our method differs from the spring-based method in that ours is position-based; thus ours can perform numerically stable simulation with larger time steps.

Particle-based methods The particle-based method for simulating fluid is popular in computer graphics applications and has been developed for enforcing fluid incompressibility, one-way and two-way solid coupling, and fluid-fluid interactions [35]. For incompressible fluids, various pressure solvers have been proposed, e.g., Weakly Compressible SPH [36], Predictive-Corrective Incompressible SPH [37], and Implicit Incompressible SPH [38]. Interactions between fluid and objects are important and have also been improved for one-way and two-way coupling with rigid bodies [39, 40, 41] and deformable objects [42]. Additionally, for fluid-fluid interactions, several methods have been proposed to generate bubbles [43], simulate fluid with significant differences in density [44], and produce the effects of diffusing, cleansing, and foaming [45].

To accelerate particle-based fluid simulations, sophisticated techniques have been presented for multi-core CPUs [46] or GPUs [47, 48]. As a different direction to efficient computation, adaptively sampled particles reduced computational cost while preserving the quality of the simulation results [49, 50, 51].

3. Proposed Method

We use position-based fluids [8] as our underlying fluid solver because we can adopt larger time steps than those used in SPH; thus, we can take full advantage of our geometric approach to maximize time steps.

We briefly summarize the fundamental formulations of position-based dynamics [9, 10] in Section 3.1 and explain our fluid solver in Section 3.2. In Section 3.3, we give details on how to simulate viscosity in our framework. We explain the handling of elasticity in Section 3.4 and give an algorithm for controlling constraints in Section 3.5. We explain thermal conduction and its effects in Section 3.6 and show the procedure of our method in Section 3.7.

3.1. Position-Based Dynamics

In position-based dynamics, an object is represented by a set of N particles and a set of scalar constraints. A particle i has its mass m_i, position $\mathbf{p}_i$, and velocity $\mathbf{v}_i$. The position of each particle is iteratively corrected by resolving the constraints, conserving the linear and angular momenta of the object. Let $\mathbf{p}$ denote the collection of particle positions $\mathbf{p}_1, \ldots, \mathbf{p}_N$, and a constraint of $\mathbf{p}$ as $C(\mathbf{p})$, which is preserved to satisfy the bilateral ($C(\mathbf{p}) = 0$) or unilateral ($C(\mathbf{p}) \geq 0$) condition.

We aim to find a correction vector $\Delta\mathbf{p}$ such that $C(\mathbf{p}+\Delta\mathbf{p}) = 0$. This constraint is approximated by

$$C(\mathbf{p} + \Delta\mathbf{p}) \approx C(\mathbf{p}) + \nabla_{\mathbf{p}} C(\mathbf{p})\Delta\mathbf{p} = 0, \tag{1}$$

and $\Delta\mathbf{p}$ is restricted in the direction of $\nabla_{\mathbf{p}} C(\mathbf{p})$ with a scaling factor λ to conserve the linear and angular momenta:

$$\Delta\mathbf{p} = \lambda \nabla_{\mathbf{p}} C(\mathbf{p}). \tag{2}$$

For a particle i, we obtain a scaling factor λ_i with Eqs. (1) and (2):

$$\lambda_i = -\frac{w_i C_i(\mathbf{p})}{\sum_j w_j \|\nabla_{\mathbf{p}_j} C_i(\mathbf{p})\|^2},$$

where $w_i = 1/m_i$, and a position correction vector is given by

$$\Delta\mathbf{p}_i = -\frac{w_i C_i(\mathbf{p})}{\sum_j w_j \|\nabla_{\mathbf{p}_j} C_i(\mathbf{p})\|^2} \nabla_{\mathbf{p}_i} C_i(\mathbf{p}). \tag{3}$$

When a constraint with the unilateral condition is true, position correction is simply skipped. Note that the dimensionalities of $C(\mathbf{p})$ and λ vary depending on the types of constraints.

3.2. Position-Based Fluids with Boundary-Handling

Since the original method of position-based fluids [8] fails to correctly estimate particle density on object surfaces due to the deficiency of particles, we improve density estimation with non-fluid particles on surfaces by modifying the method of Akinci et al. [41] for position-based fluids.

In position-based fluids, constraints are imposed on each particle of a fluid. For a particle i with its density ρ_i, the dimensionless density constraint with the bilateral condition is defined as

$$C_{\mathrm{dens},i}(\mathbf{p}) = \frac{\rho_i}{\rho_0} - 1 = 0, \tag{4}$$

where ρ_0 is the rest density of the fluid, and ρ_i is estimated with the summation approach commonly used in SPH, taking non-fluid particles into account [41]:

$$\rho_i = \sum_j m_j W_{ij} + \sum_k \frac{\rho_0}{\delta_k} W_{ik}, \tag{5}$$

where j denotes a *neighboring* fluid particle of i, k a *neighboring* non-fluid particle, W_{ij} the short for a kernel $W(\mathbf{p}_i - \mathbf{p}_j, h)$ with a kernel radius h, and the number density $\delta_i = \sum_k W_{ik}$. The gradient of constraint Eq. (4) with respect to particle l is given by

$$\nabla_{\mathbf{p}_l} C_{\mathrm{dens},i}(\mathbf{p}) = \begin{cases} \frac{1}{\rho_0}\sum_j m_j \nabla W_{ij} + \sum_k \frac{1}{\delta_k}\nabla W_{ik} & \text{if } l = i, \\ -\frac{m_j}{\rho_0}\nabla W_{ij} & \text{if } l = j, \\ -\frac{1}{\delta_k}\nabla W_{ik} & \text{if } l = k, \end{cases} \tag{6}$$

and a density scaling factor $\lambda_{\mathrm{dens},i}$ is computed by

$$\lambda_{\mathrm{dens},i} = -\frac{w_i C_{\mathrm{dens},i}(\mathbf{p})}{\sum_l w_l \|\nabla_{\mathbf{p}_l} C_{\mathrm{dens},i}(\mathbf{p})\|^2}. \tag{7}$$

To handle the particle clustering induced by the negative pressures around free surfaces due to the deficiency of particles, Macklin and Müller [8] introduced an artificial pressure term:

$$s_{\mathrm{corr},i} = -k_s h^2 \left(\frac{W_{ij}}{W(\mathbf{q}, h)}\right)^4, \tag{8}$$

where k_s is a parameter to control the effect of artificial pressure, and $\mathbf{q}$ is a vector whose norm is less than h. In summary, with Eqs. (2), (6), (7), and (8), the density correction vector is computed by

$$\Delta\mathbf{p}_{\mathrm{dens},i} = \frac{1}{\rho_0}\sum_j m_j(\lambda_{\mathrm{dens},i} + \lambda_{\mathrm{dens},j} + s_{\mathrm{corr},i})\nabla W_{ij} + \sum_k \frac{1}{\delta_k}(2\lambda_{\mathrm{dens},i} + s_{\mathrm{corr},i})\nabla W_{ik}.$$

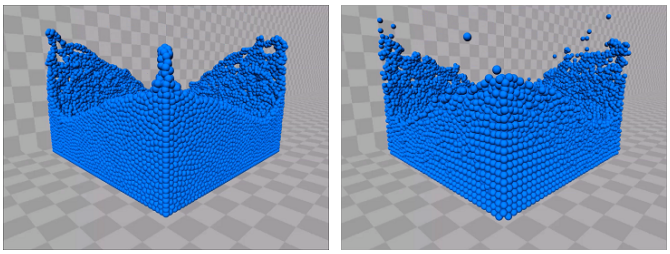
Additionally, XSPH viscosity [23] is used for coherent motions of particles with $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$:

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \epsilon_a \sum_j \frac{m_j}{\rho_j}\mathbf{v}_{ji} W_{ij} + \epsilon_b \sum_k \frac{1}{\delta_k}\mathbf{v}_{ki} W_{ik},$$

where ϵ_a and ϵ_b are parameters for viscous effects.

Since our aim is to simulate viscous fluids, we do not use vorticity confinement unlike in the original work [8]. Note that by taking non-fluid boundary particles into the density estimation in Eq. (5), they are also incorporated into the relevant formulations without any need of special treatments; therefore our method ensures appropriate handling of particles on object surfaces.

Figure 2 compares the original position-based fluids adopting level set boundary-handling [8] with our method. While the original method [8] causes the artifacts of particle stacking along the vertical edge and unnatural particle aggregations along the horizontal edges caused by erroneously underestimated particle density, our method can generate natural fluid motions thanks to the proposed boundary-handling scheme.



(a) Position-based fluids [8] (b) Our method

Figure 2: Comparison of dam break simulation in a virtual box.

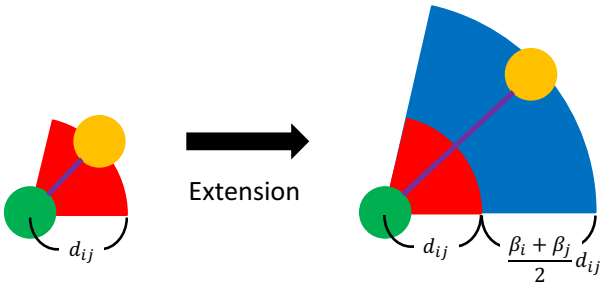


Figure 3: Illustration of extending interparticle distance d_{ij} for viscosity constraints. When two particles (green and orange) are connected, orange particle can move within red area in left figure. After interparticle distance d_{ij} is updated with extension coefficients β_i and β_j, orange particle can move within red and blue areas in right figure.

3.3. Viscosity

As described in Section 1, the current mainstream for simulating viscous fluids is based on the Navier-Stokes equations. To describe viscosity in the equations, several viscosity terms have been proposed [5, 6, 19]. However, their explicit schemes fail to simulate viscous fluids under high spatial resolution in a numerically stable manner due to the stability condition caused by diffusion equations. To avoid this problem, we herein propose a particle-based method with geometric constraints.

3.3.1. Viscosity Constraint Mechanism

We approximate the effect of viscosity using constraints between two particles connected to each other (hereafter, we call these particles *connected* particles). Since particle velocities are updated with position differences between consecutive simulation steps in position-based dynamics, we can smooth particle velocities by confining motions of connected particles within certain distances, and this smoothing effect is almost equivalent to dissipating particle velocities to neighboring particles, as described with the Laplacian operator. Specifically, we preserve interparticle distances with constraints that directly correct particle positions for the smoothing effect.

We dynamically generate, modify, and delete constraints between particles in the simulation. If the distance of two particles is less than the kernel radius h, and there is no constraint between them, we generate a new constraint. Since positions of particles are corrected to keep their interparticle distance, we modify a constraint by extending the distance so that particles can move in the opposite direction from the other. We delete the constraint between particles when the distance of the connected particles is larger than H (we set $H = 2h$). After we explain elasticity in Section 3.4, we give an algorithm for controlling constraints of viscosity and elasticity in Section 3.5.

3.3.2. Viscosity Constraint

To restrict particle motions when particles are separating from each other, we correct particle positions based on the viscosity constraint $C_{\text{visc}}(\mathbf{p}_i, \mathbf{p}_j)$ with the unilateral condition, defined with the positions of two particles and their interparticle distance d_{ij} as

$$C_{\text{visc}}(\mathbf{p}_i, \mathbf{p}_j) = d_{ij} - \|\mathbf{p}_{ij}\| \geq 0, \tag{9}$$

where $\mathbf{p}_{ij} = \mathbf{p}_i - \mathbf{p}_j$, and d_{ij} is initialized as the distance of the two particles when the constraint is generated. We adopt the unilateral condition for the constraint because a constraint with the bilateral condition tends to disturb the homogeneous particle distributions induced by the artificial pressures. Viscosity constraints handle only the expansion of fluid volumes, and density constraints address the compression of the volumes. Note that the dimensionality of the viscosity constraint differs from that of the density constraint. If Eq. (9) does not hold, we correct the positions of connected particles using Eq. (3) with a scaling factor s_v ($0 \leq s_v \leq 1$) to maintain their interparticle distance:

$$\Delta\mathbf{p}_{\text{visc},i} = -\frac{s_v w_i}{w_i + w_j}(\|\mathbf{p}_{ij}\| - d_{ij})\frac{\mathbf{p}_{ij}}{\|\mathbf{p}_{ij}\|}, \tag{10}$$

$$\Delta\mathbf{p}_{\text{visc},j} = +\frac{s_v w_j}{w_i + w_j}(\|\mathbf{p}_{ij}\| - d_{ij})\frac{\mathbf{p}_{ij}}{\|\mathbf{p}_{ij}\|}. \tag{11}$$

Since Eq. (9) is likely to be exceedingly strict for modifying the viscosity constraint, we subtract the term αd_{ij} to weaken the condition as

$$\begin{aligned} D_{\text{visc}}(\mathbf{p}_i, \mathbf{p}_j) &= C_{\text{visc}}(\mathbf{p}_i, \mathbf{p}_j) - \alpha d_{ij} \\ &= d_{ij} - \|\mathbf{p}_{ij}\| - \alpha d_{ij} \geq 0. \end{aligned} \tag{12}$$

If Eq. (12) does not hold, we extend an interparticle distance to enable connected particles to slightly separate from the other (see Figure 3):

$$d_{ij} \leftarrow d_{ij} + \frac{\beta_i + \beta_j}{2} d_{ij}, \tag{13}$$

where β_i and β_j are extension coefficients that control the variation in the interparticle distance. Note that we modify the extension coefficient, not viscosity value, and a high (low) extension coefficient is equivalent to low (high) viscosity. Our method can produce motions of fluids with different viscosity values using different extension coefficients, as shown in Figure 4.

3.4. Elasticity

Elasticity, which differs from viscosity, is a characteristic disposition that some materials exhibit. According to Zhou et

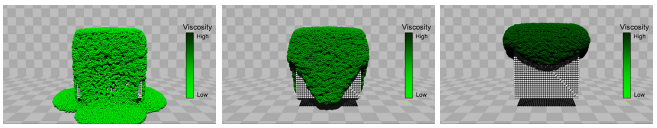


Figure 4: Motion comparison of fluids with different extension coefficients. β = 1.005, 1.003, and 1.001 from left to right. Particles are colored according to their viscosity values (low viscosity: light green, and high viscosity: dark green).

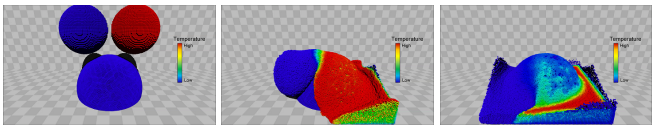


Figure 5: Two dropped fluid balls mix with each other being cooled by cold boundary particles in a virtual box on the ground. Each fluid particle has a different extension coefficient computed based on particle's temperature. Particles are colored according to their temperatures (low temperature: blue, mid-temperature: green, and high temperature: red).

al. [52], an elastic force term requires small time steps to simulate elastic materials in a numerically stable manner. To avoid this condition, Zhou et al. [52] relied on an implicit method and successfully simulated elastic materials. However, the deformations of the materials are restricted to some extent because they used a Cauchy tensor, ignoring a second-order term, to build a linear system in an implicit manner. Unlike their method, we still depend on a position-based method to simulate highly deformable elastic materials.

As the basic concept regarding the handling of elasticity is to correct positions of particles based on constraints, which are the same as viscosity constraints, we also use Eq. (9) as elasticity constraints. For viscous effects, we extend the interparticle distance d_{ij}, which is a reference distance when we correct particle positions, by using Eq. (13). For elasticity, however, we do *not* extend the interparticle distance d_{ij}; therefore, we can generate motions of deformable objects with density constraints, which address fluid compression, by restoring the original configurations of particles, as in position-based dynamics [9, 10]. We set initial constraints to determine the shape of objects when they are created. If we always correct particle positions using Eqs. (10) and (11) without generating, modifying, and deleting constraints, our method ensures the reproduction of elastic motions of objects.

3.5. Constraint Control Algorithm

We summarize control steps for generating, modifying, and deleting constraints in Algorithm 1. Note that we use the *viscous* label for viscous fluid particles and *elastic* for particles in elastic materials.

Algorithm 1 Constraint control

```
for all fluid particle i do
    for all connected particle j do
        if at least either i or j is viscous then
            if H ≤ ||p_ij|| then
                delete the constraint
            if not satisfy Eq. (12) then
                modify the constraint
    for all neighboring particle j do
        if at least either i or j is viscous then
            if there is no constraint then
                generate a new constraint
```

3.6. Thermal Conduction

The characteristics of materials can change depending on their temperature. For example, high-temperature materials tend to be less viscous and exhibit liquid-like motions, while low-temperature materials are highly viscous and can be quasi-rigid or -elastic from fluid through phase transition. To reproduce these phenomena, we incorporate thermal conduction into our framework by computing the temperature of particles on the basis of thermal diffusion equations. Specifically, we improve the method proposed by Cleary and Monaghan [53] to handle heat transfer on object surfaces with non-fluid boundary particles:

$$T_i \leftarrow T_i + \frac{1}{\rho_i}\sum_j \frac{2c_i c_j}{c_i + c_j}\frac{m_j}{\rho_j} T_{ji}\nabla^2 W_{ij} + \frac{1}{\rho_i}\sum_k \frac{2c_i c_k}{c_i + c_k}\frac{1}{\delta_k} T_{ki}\nabla^2 W_{ik},$$

where $T_{ij} = T_i - T_j$ and $c = s_c\rho_0 h^2$ with a parameter s_c to control the speed of heat propagation.

We then change values in the extension coefficients to vary the viscosity of fluid. Although there are many models for relating temperature and viscosity, we use a simple linear interpolation to compute extension coefficient β_i with particle temperature for our aim of fast simulation:

$$\beta_i = \begin{cases} \beta_b & T_b < T_i, \\ \beta_a + (\beta_b - \beta_a)\frac{T_i - T_a}{T_b - T_a} & T_a < T_i \le T_b, \\ \beta_a & T_i \le T_a, \end{cases}$$

where β_a and β_b ($\beta_a < \beta_b$) are minimum and maximum values of β_i, respectively, and T_a and T_b ($T_a < T_b$) are lower and upper thresholds of T_i, respectively. This simple model is sufficient to generate plausible results, as shown in Figure 5, where a cold ball (blue) and a hot one (red) are dropped onto a solid cold ball, and high-temperature particles flow quickly while low-temperature particles do so slowly. In addition to the changes in the extension coefficients, we alter the state of particles to simulate the phase transition in materials by labeling the particles as *elastic* (*viscous*) if their temperatures are lower (higher) than a threshold temperature T_c ($< T_a$).

Our method can handle various types of objects, such as viscous fluids and elastic materials with phase transition and changes in physical properties, due to the thermal conduction in a unified position-based dynamics framework.

3.7. Procedure of Our Method

Algorithm 2 gives a full-fledged procedure of the steps performed in our method. An object $\mathcal{S}_n$ has $\mathcal{M}_n$ constraints, denoted as $C_{n,1}, \ldots, C_{n,\mathcal{M}_n}$, which are generated, modified, and deleted using Algorithm 1 (line 11). We iterate the position correction step until its iteration reaches a specified count. Note that the viscosity constraints are solved with density constraints in parallel to satisfy both types of constraints as much as possible because the effect of one type of constraint can be ignored if we solve one side prior to the other.

Algorithm 2 Procedure of our method

1: initialize all variables
2: **while** animating **do**
3: **for all** particle i **do**
4: **if** fluid particle **then**
5: apply XSPH viscosity
6: apply external forces $\mathbf{v}_i \leftarrow \mathbf{v}_i + \Delta t \mathbf{f}_{\text{ext}}(\mathbf{x}_i)$
7: predict position $\mathbf{p}_i \leftarrow \mathbf{x}_i + \Delta t \mathbf{v}_i$
8: update temperature $T_i \leftarrow T_i + \Delta T_i$
9: **for all** particle i **do**
10: find neighboring particles at $\mathbf{p}_i$
11: control constraints $\mathcal{S}_1(C_{1,1}, \ldots, C_{1,\mathcal{M}_1}), \ldots$
12: **while** correcting particle positions **do**
13: **for all** active non-fluid particle i **do**
14: compute δ_i
15: **for all** fluid particle i **do**
16: compute λ_i and $s_{\text{corr},i}$
17: **for all** fluid particle i **do**
18: compute $\Delta\mathbf{p}_{\text{dens},i}$ and $\Delta\mathbf{p}_{\text{visc},i}$
19: **for all** fluid particle i **do**
20: correct position $\mathbf{p}_i \leftarrow \mathbf{p}_i + \Delta\mathbf{p}_{\text{dens},i} + \Delta\mathbf{p}_{\text{visc},i}$
21: **for all** particle i **do**
22: update velocity $\mathbf{v}_i \leftarrow (\mathbf{p}_i - \mathbf{x}_i)/\Delta t$
23: update position $\mathbf{x}_i \leftarrow \mathbf{p}_i$
24: compute ΔT_i

4. Results

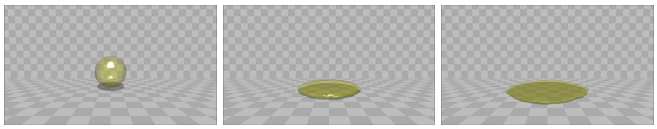
We implemented our algorithm in C++ and parallelized it with OpenMP 2.0. We used constant time steps in each scene and only six iterations of position correction for fast simulation. In our simulations, parameters were empirically adjusted according to the scenes, and non-fluid boundary particles used the extension coefficient of their connected particles. All the scenes were executed on a PC with a 4-core Intel Core i7 3.50 GHz CPU and RAM 16.0 GB. We used six threads in total and obtained the accelerated simulation performance by a factor from two to four. Fluid surfaces were extracted with anisotropic kernels [54]. All the results were rendered with a free open-source raytracer POV-Ray 3.7, and off-line rendering required about 60 seconds per frame on average to generate Figure 1. The accompanying video shows several results produced with our method and a previous method which uses explicit viscosity integration for comparison. We tabulate the simulation conditions and performances of the results in Table 1.

Table 1: Simulation statistics.

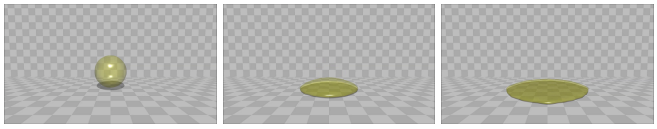
Figure #	# of particles fluid/non-fluid	Time / frame (s)
1	up to 307.2k/111.4k	47.1
2 (a)	24.8k/0.0	0.5
2 (b)	24.8k/6.8k	0.6
4	27.6k/59.5k	1.8
5	364.1k/126.1k	40.5
6 (a)	7.8k/7.7k	0.5
6 (b)	7.8k/7.7k	82.7
6 (c)	7.8k/7.7k	0.5
7	up to 23.4k/34.0k	2.8
8	up to 92.1k/34.0k	6.3
9	up to 269.4k/111.5k	36.2
10	65.2k/63.7k	5.8
11	up to 33.1k/35.7k	1.9
12	15.9k/75.7k	2.4

4.1. Comparison

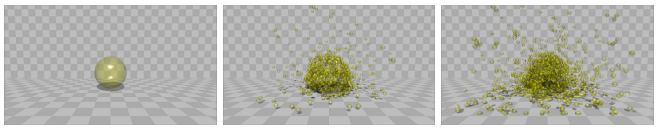
To prove that our method can use larger time steps than those adopted in SPH with explicit viscosity integration, we used a simple scenario in which a viscous fluid ball was dropped onto the ground. As a method for comparison, we used Predictive-Corrective Incompressible SPH [37], instead of position-based fluids, with Laplacian form of viscosity because the combination of position-based fluids and Laplacian form of viscosity could not generate plausible viscous fluid behavior. In force-based methods, using smaller time steps contributes to producing more accurate simulation results, yet this fact is inapplicable to position-based methods. As noted in the first paper on position-based dynamics proposed by Müller et al. [9], time step size affects simulation results. Specifically, the position-based dynamics method with smaller time steps generates stiffer object motions, while using larger time steps leads to generating softer and more deformable object motions. Therefore, position-based fluids with extremely small time steps, which are needed for Laplacian form of viscosity, did not produce highly deformable motions of fluid. Although the comparison of the previous method and ours is not completely fair, the viscosity term is a dominant factor for determining time steps in the scene, and this comparison is intended to show that our method can use much larger time steps than the traditional method to reduce the number of simulation loops each of which includes a time-consuming neighbor search step. Our method required $\Delta t = 9.06 \times 10^{-4}$ s to produce plausible viscous fluid motions (Figure 6 (a)), while the previous method required $\Delta t = 5.41 \times 10^{-6}$ s to generate a similar result (Figure 6 (b)). As shown in Figure 6 (c), we could not generate a plausible result using the previous method with the same time step used in Figure 6 (a). Our method (Figure 6 (a)) was able to use 167.47 times larger time steps than those adopted with the previous method (Figure 6 (b)); therefore our method could produce visually plausible viscous fluid motions with much lower computational cost compared to the previous method.



(a) Our method ($\Delta t = 9.06 \times 10^{-4}$ s)



(b) Laplacian form ($\Delta t = 5.41 \times 10^{-6}$ s)



(c) Laplacian form ($\Delta t = 9.06 \times 10^{-4}$ s)

Figure 6: Comparison of resulting motions.

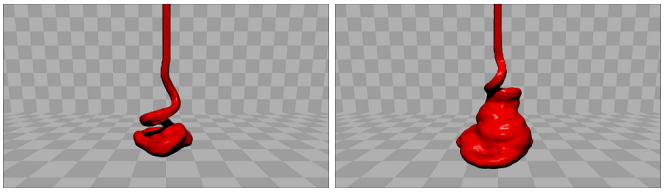


Figure 7: Viscous thread coiling reproduced with our method.

4.2. Coiling and Buckling

Figures 7 and 8 demonstrate coiling and buckling phenomena, respectively, which highly viscous fluids exhibit. In these scenes, a viscous material is dropped onto the ground, while particles are continuously added on top of the material. Our method can generate characteristic and complicated coiling and buckling phenomena despite the approximation of viscous fluid dynamics with position-based constraints.

4.3. Variable Viscosity

Figure 9 provides an example with materials of different viscosity values. Several slime lumps with different viscosity values are successively dropped onto a solid bunny, mixing with each other. Since each particle has its own parameters, spatial variation in viscosity can be easily achieved.

4.4. Boundary-Handling

Figure 10 demonstrates that our boundary-handling scheme is applicable to a moving solid object with non-fluid boundary particles used for the positional constraints. In this scene, a viscous lump drops onto a rotating bar, avoids the bar by dividing into two parts, and finally falls onto the ground.

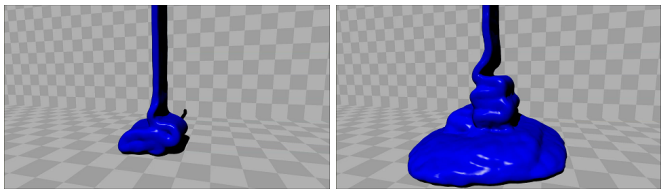


Figure 8: Viscous sheet buckling reproduced with our method.

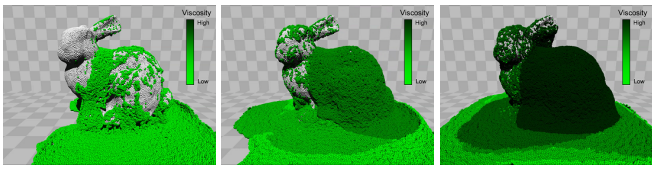


Figure 9: Several slime lumps with different viscosity values successively dropping onto a solid bunny. Particles are colored according to their viscosity values (low viscosity: light green, and high viscosity: dark green).

4.5. Phase Transition

Figure 11 shows the results of a hot viscous fluid poured onto a cold slope. First, the fluid flows on the slope due to low viscosity, while the fluid hardens and finally congeals through phase transition as it is cooled by the cold particles of the slope. Figure 12 demonstrates the phase transition in a single object. After an elastic ball runs down a cold slope, the ball bounces a few times, sticks to a hot wall, and starts melting. The combination of viscosity and elasticity constraints can work well through phase transition and generate plausible results.

5. Discussions and Limitations

The configuration of particles is highly relevant to the quality of fluid surfaces due to the dependence of our surface reconstruction on the particles. Since each particle in position-based fluids has fewer neighboring ones than those in SPH simulations, individual particles significantly affect neighboring ones. Consequently, a particle around a free surface, which originally has fewer neighboring particles, is easily pushed toward the surface, causing local irregularities and giving rise to bumpy surfaces. This problem can be alleviated with a sophisticated surface reconstruction method [55].

Even if viscosity constraints are of no effect, fluid particles on object surfaces are likely to unnaturally adhere to non-fluid ones to satisfy the density constraints by compensating for low particle density. Although our boundary-handling scheme is necessary to specify boundary conditions with constraints, it cannot fully prevent fluid particles from sticking to non-fluid ones. To address this problem, the use of air particles [56] at the significant sacrifice of computational efficiency will help us to precisely estimate the particle density on surfaces.

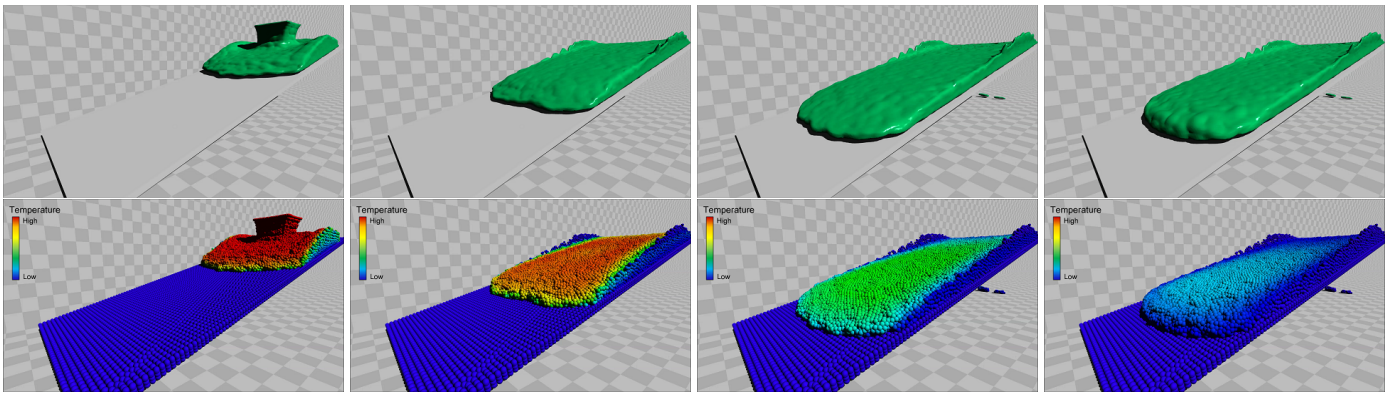


Figure 11: Hot viscous fluid poured onto a cold slope. Lower images are corresponding particle representations, where particles are colored according to their temperatures (low temperature: blue, mid-temperature: green, and high temperature: red).

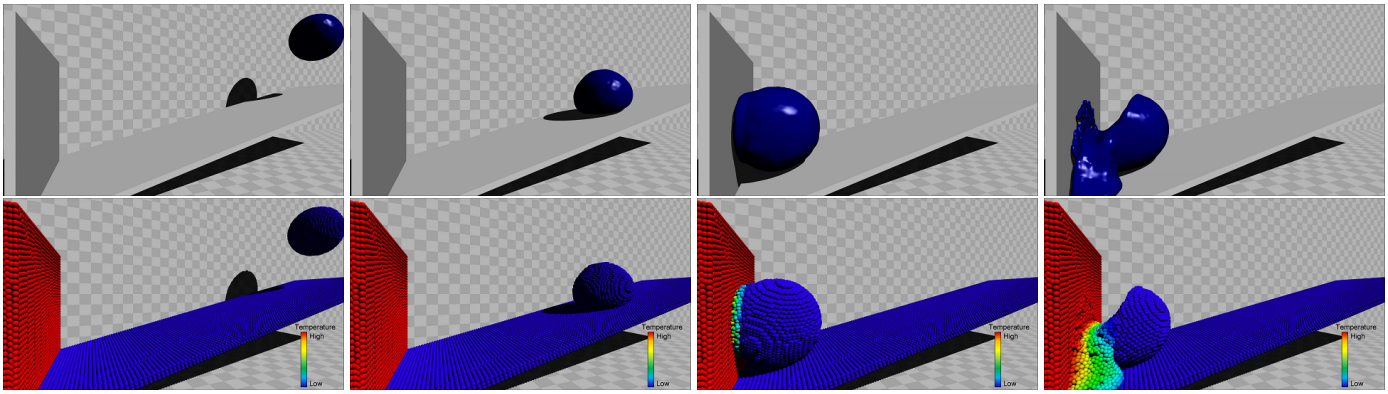


Figure 12: Elastic ball running down a cold slope. Lower images are corresponding particle representations, where particles are colored according to their temperatures (low temperature: blue, mid-temperature: green, and high temperature: red).

Our method can encompass a broad range of viscosity: from almost inviscid to highly viscous fluids (see Figures 4 and 9). Fortunately, the strength of the viscosity constraints is virtually irrelevant to the required iterations for convergence. We can also use the same time steps regardless of the strength. However, since time steps can affect the viscosity of fluid, we need to choose appropriate time steps to generate desirable viscous fluid motions. Finding appropriate time steps would be difficult due to the nonlinearity of resolving the constraints and the lack of physical grounds, and this point will be investigated further as an important topic for future work.

6. Conclusions

We extended the previous paper [11] and proposed a particle-based Lagrangian method for simulating viscous fluids using position-based dynamics. Fluid volumes are discretized by particles that interact with other neighboring ones under position-based constraints, approximating the dynamics of viscous fluids. Hence, our method allowed for the use of larger time steps than those adopted in SPH methods with explicit viscosity integration, while enabling us to reproduce plausible motions of fluids with different properties in a unified framework. An associated boundary-handling scheme for position-based fluids appropriately addressed boundary conditions for the constraints. Moreover, we incorporated handling of elastic materials into our framework and added thermal conduction to simulate changes in particle properties and phase transition in the materials. We proved the effectiveness and versatility of our method with several examples.

Acknowledgements

This work has been partly supported by Japan Society for the Promotion of Science under Grant-in-Aid for Scientific Research (A) No. 26240015. We would like to thank Nobuyuki Umetani from Disney Research and anonymous reviewers for their valuable suggestions and comments.

References

[1] Carlson M, Mucha PJ, Van Horn III RB, Turk G. Melting and flowing. In: Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2002, p. 167–74.

[2] Rasmussen N, Enright D, Nguyen D, Marino S, Sumner N, Geiger W, et al. Directable photorealistic liquids. In: Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2004, p. 193–202.

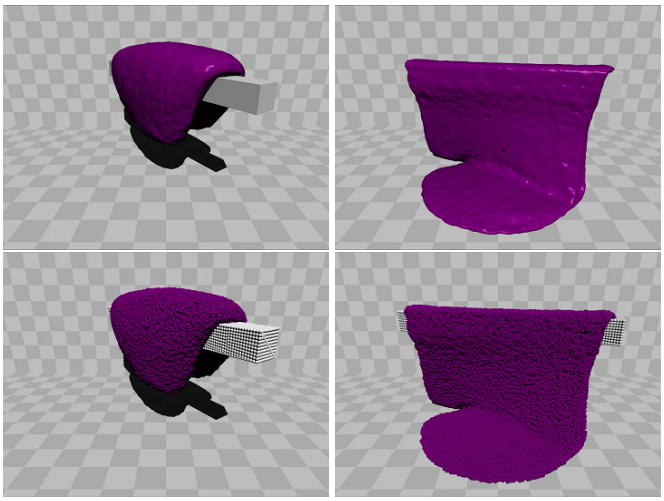


Figure 10: Viscous lump dropping onto a rotating bar. Lower images are corresponding particle representations.

[3] Batty C, Bridson R. Accurate viscous free surfaces for buckling, coiling, and rotating liquids. In: Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2008, p. 219–28.

[4] Batty C, Houston B. A simple finite volume method for adaptive viscous liquids. In: Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2011, p. 111–8.

[5] Paiva A, Petronetto F, Lewiner T, Tavares G. Particle-based viscoplastic fluid/solid simulation. Computer-Aided Design 2009;41(4):306–14.

[6] Müller M, Charypar D, Gross M. Particle-based fluid simulation for interactive applications. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2003, p. 154–9.

[7] Charney JG, Fjortoft R, Von Neumann J. Numerical integration of the barotropic vorticity equation. Tellus 1950;2(4):237–54.

[8] Macklin M, Müller M. Position based fluids. ACM Transactions on Graphics 2013;32(4):1–5.

[9] Müller M, Heidelberger B, Hennix M, Ratcliff J. Position based dynamics. Journal of Visual Communication and Image Representation 2007;18(2):109–18.

[10] Bender J, Müller M, Otaduy MA, Teschner M. Position-based methods for the simulation of solid objects in computer graphics. In: EUROGRAPHICS 2013 State of the Art Reports. 2013, p. 1–22.

[11] Takahashi T, Fujishiro I. Accelerated viscous fluid simulation using position-based constraints. In: Proceedings of CAD/Graphics 2013. 2013, p. 260–7.

[12] Bargteil AW, Wojtan C, Hodgins JK, Turk G. A finite element method for animating large viscoplastic flow. ACM Transactions on Graphics 2007;26(3).

[13] Wojtan C, Turk G. Fast viscoelastic behavior with thin features. ACM Transactions on Graphics 2008;27(3):1–8.

[14] Wicke M, Ritchie D, Klingner BM, Burke S, Shewchuk JR, O'Brien JF. Dynamic local remeshing for elastoplastic simulation. ACM Transactions on Graphics 2010;29:1–11.

[15] Clausen P, Wicke M, Shewchuk JR, O'Brien JF. Simulating liquids and solid-liquid interactions with Lagrangian meshes. ACM Transactions on Graphics 2013;32(2):1–15.

[16] Bergou M, Audoly B, Vouga E, Wardetzky M, Grinspun E. Discrete viscous threads. ACM Transactions on Graphics 2010;29(4):1–10.

[17] Batty C, Uribe A, Audoly B, Grinspun E. Discrete viscous sheets. ACM Transactions on Graphics 2012;31(4):1–7.

[18] Desbrun M, Gascuel MP. Smoothed particles: A new paradigm for animating highly deformable bodies. In: Proceedings of the Eurographics Workshop on Computer Animation and Simulation. 1996, p. 61–76.

[19] Monaghan JJ. Smoothed particle hydrodynamics. Annual Review of Astronomy and Astrophysics 1992;30:543–74.

[20] Stora D, olivier Agliati P, paule Cani M, Neyret F, dominique Gascuel J. Animating lava flows. In: Graphics Interface. 1999, p. 203–10.

[21] Solenthaler B, Schläfli J, Pajarola R. A unified particle model for fluid solid interactions: Research articles. Computer Animation and Virtual Worlds 2007;18(1):69–82.

[22] Becker M, Ihmsen M, Teschner M. Corotated SPH for deformable solids. In: Proceedings of Eurographics Workshop on Natural Phenomena. 2009, p. 27–34.

[23] Monaghan JJ. On the problem of penetration in particle methods. Journal of Computational Physics 1989;82(1):1–15.

[24] Mao H, Yang YH. Particle-based non-newtonian fluid animation with heating effects. Tech. Rep.; University of Alberta; 2006.

[25] Chang Y, Bao K, Liu Y, Zhu J, Wu E. A particle-based method for viscoelastic fluids animation. In: Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology. 2009, p. 111–7.

[26] Rafiee A, Manzari M, Hosseini M. An incompressible SPH method for simulation of unsteady viscoelastic free-surface flows. International Journal of Non-Linear Mechanics 2007;42(10):1210 –23.

[27] Gerszewski D, Bhattacharya H, Bargteil AW. A point-based method for animating elastoplastic solids. In: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2009, p. 133–8.

[28] Takamatsu K, Kanai T. A fast and practical method for animating particle-based viscoelastic fluids. The International Journal of Virtual Reality 2011;10:29–35.

[29] Dagenais F, Gagnon J, Paquette E. A prediction-correction approach for stable SPH fluid simulation from liquid to rigid. In: Proceedings of the Computer Graphics International 2012. 2012,.

[30] Miller G, Pearce A. Globular dynamics: A connected particle system for animating viscous fluids. Computers and Graphics 1989;13(3):305–9.

[31] Terzopoulos D, Platt J, Fleischer K. Heating and melting deformable models. Journal of Visualization and Computer Animation 1991;2:68–73.

[32] Steele K, Cline D, Egbert PK, Dinerstein J. Modeling and rendering viscous liquids. Computer Animation and Virtual Worlds 2004;15(3-4):183–92.

[33] Tamura N, Nakaguchi T, Tsumura N, Miyake Y. Spring-bead animation of viscoelastic materials. IEEE Computer Graphics and Applications 2007;27(6):87–93.

[34] Clavet S, Beaudoin P, Poulin P. Particle-based viscoelastic fluid simulation. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2005, p. 219–28.

[35] Ihmsen M, Orthmann J, Solenthaler B, Kolb A, Teschner M. SPH fluids in computer graphics. In: EUROGRAPHICS 2014 State of the Art Reports. 2014, p. 21–42.

[36] Becker M, Teschner M. Weakly compressible SPH for free surface flows. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2007, p. 209–17.

[37] Solenthaler B, Pajarola R. Predictive-corrective incompressible SPH. ACM Transactions on Graphics 2009;28(3):1–6.

[38] Ihmsen M, Cornelis J, Solenthaler B, Horvath C, Teschner M. Implicit incompressible SPH. IEEE Transactions on Visualization and Computer Graphics 2014;20(3):426–35.

[39] Becker M, Tessendorf H, Teschner M. Direct forcing for Lagrangian rigid-fluid coupling. IEEE Transactions on Visualization and Computer Graphics 2009;15(3):493–503.

[40] Ihmsen M, Akinci N, Gissler M, Teschner M. Boundary handling and adaptive time-stepping for PCISPH. In: Proceedings of 10th Workshop on Virtual Reality Interaction and Physical Simulation. 2010, p. 79–88.

[41] Akinci N, Ihmsen M, Akinci G, Solenthaler B, Teschner M. Versatile rigid-fluid coupling for incompressible SPH. ACM Transactions on Graphics 2012;31(4):1–8.

[42] Akinci N, Cornelis J, Akinci G, Teschner M. Coupling elastic solids with smoothed particle hydrodynamics fluids. Computer Animation and Virtual Worlds 2013;24(3-4):195–203.

[43] 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. 2005, p. 237–44.

[44] Solenthaler B, Pajarola R. Density contrast SPH interfaces. In: Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Com-

puter Animation. 2008, p. 211–8.
[45] Orthmann J, Hochstetter H, Bader J, Bayraktar S, Kolb A. Consistent surface model for SPH-based fluid transport. In: Proceedings of the 2013 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2013, p. 95–103.
[46] Ihmsen M, Akinci N, Becker M, Teschner M. A parallel SPH implementation on multi-core CPUs. Computer Graphics Forum 2011;30(1):99–112.
[47] Harada T, Koshizuka S, Kawaguchi Y. Smoothed particle hydrodynamics on GPUs. In: Proceedings of Computer Graphics International. 2007, p. 63–70.
[48] Goswami P, Schlegel P, Solenthaler B, Pajarola R. Interactive SPH simulation and rendering on the GPU. In: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2010, p. 55–64.
[49] Adams B, Pauly M, Keiser R, Guibas LJ. Adaptively sampled particle fluids. ACM Transactions on Graphics 2007;26(3).
[50] Solenthaler B, Gross M. Two-scale particle simulation. ACM Transactions on Graphics 2011;30(4):1–8.
[51] Orthmann J, Kolb A. Temporal blending for adaptive SPH. Computer Graphics Forum 2012;31(8):2436–49.
[52] Zhou Y, Lun Z, Kalogerakis E, Wang R. Implicit integration for particle-based simulation of elasto-plastic solids. Computer Graphics Forum 2013;32(7):215–23.
[53] Cleary PW, Monaghan JJ. Conduction modelling using smoothed particle hydrodynamics. Journal of Computational Physics 1999;148(1):227–64.
[54] Yu J, Turk G. Reconstructing surfaces of particle-based fluids using anisotropic kernels. ACM Transactions on Graphics 2013;32(1):1–12.
[55] Ando R, Thürey N, Wojtan C. Highly adaptive liquid simulations on tetrahedral meshes. ACM Transactions on Graphics 2013;32(4):1–10.
[56] Schechter H, Bridson R. Ghost SPH for animating water. ACM Transactions on Graphics 2012;31(4):1–8.