CGI2014 manuscript No.
(will be inserted by the editor)

A Velocity Correcting Method for Volume Preserving Viscoelastic Fluids

Tetsuya Takahashi · Issei Fujishiro · Tomoyuki Nishita

Abstract We propose a new particle-based method for simulating viscoelastic fluids which preserve their volumes. Our method achieves the volume preservation by enforcing the incompressibility of fluid, while correcting particle velocities to approximate the dynamics of viscoelastic fluids without disturbing computations for the incompressible flow. We offer three schemes for correcting particle velocities. The first scheme employs Shape Matching proposed by Müller et al. to derive appropriate transformations of particle sets. The second computes attraction forces on the basis of Hooke's law to restrict particle motions. The third utilizes Position-Based Dynamics to restore the original relations of particle positions. The first scheme enables smooth transfers of deformation waves, the second is intuitive and simple, and the third is easy to tune parameters. We demonstrate that our method can preserve fluid volumes while generating plausible viscoelastic motions.

Keywords Fluid simulation · Viscoelasticity · Volume preservation · Velocity correction

1 Introduction

Viscoelastic fluids are common in our everyday lives, as we often see examples of these materials: egg white, gels, mucus, gelatin, and toothpaste. Due to the ubiquity of viscoelastic materials, simulating their sticky and bouncy behaviors has been required for feature films and video games.

Preserving fluid volumes is one of the most important aspects to simulate plausible fluid motions. For example, a lump of viscoelastic fluid on a flat solid surface when being compressed does not horizontally spread unless the volume of the lump is sufficiently preserved. In the computer graphics community, there are several known simulation methods for viscoelastic fluids which preserve fluid volumes using Eulerian grids [14] or Lagrangian meshes [2,42,41,9], yet we herein focus on the Lagrangian particle-based approach due to its conceptual simplicity.

In the literature of particle-based methods, combinations of *Smoothed Particle Hydrodynamics* (SPH) [25] and a geometrically-motivated method have been proposed to simulate viscoelastic fluids efficiently and robustly [10,37]. However, these previous combinations spoil the fast convergence rate of state-of-the-art fluid solvers, e.g., *Predictive-Corrective Incompressible SPH* [32] (PCISPH) and *Implicit Incompressible SPH* [17] (IISPH), which can preserve fluid volumes well by enforcing the incompressibility of fluid. Takamatsu and Kanai [37] combined SPH with a variant of *Shape Matching* [27] (SM), and interpolated two velocities derived from both methods to fast and robustly simulate viscoelastic fluids. However, since the variant of SM does not preserve the object volumes, the combined method in [37] suffers from the loss of fluid volumes. A prediction-relaxation scheme proposed by Clavet et al. [10] directly corrects particle positions based on various factors such as collision responses and spring-based forces, yet this scheme changes sets of neighboring particles and interparticle distances. Thus, their prediction-relaxation

Tetsuya Takahashi
UEI Research / Keio University
E-mail: tetsuya.takahashi@uei.co.jp

Issei Fujishiro
Keio University
E-mail: fuji@ics.keio.ac.jp

Tomoyuki Nishita
UEI Research / Hiroshima Shudo University
E-mail: tomoyuki.nishita@uei.co.jp

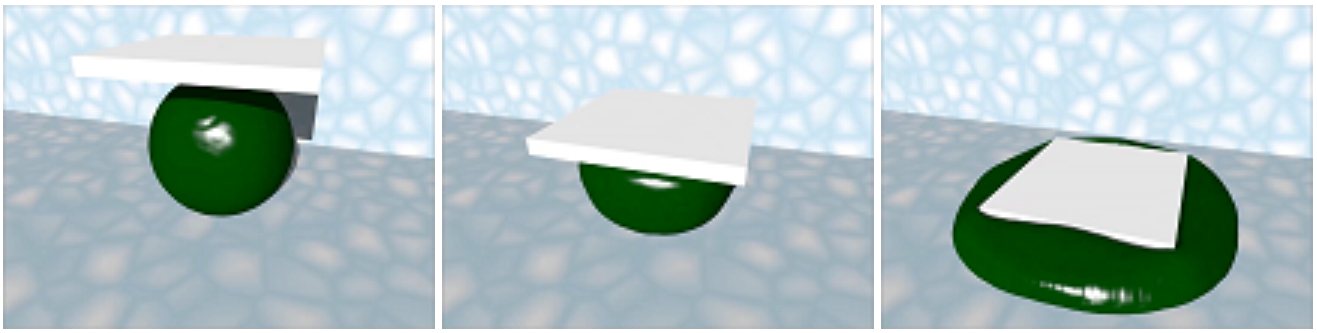


Fig. 1 A sphere of viscoelastic fluid compressed by a moving solid along a fixed path. Horizontally spreading motions are achieved due to the small volume loss of the material less than 1.0% by making use of position-based velocity corrections.

scheme disturbs the computations for the incompressible flow.

In this paper, we therefore propose a new particle-based method for simulating viscoelastic fluids which preserve their volumes. Our method utilizes velocity corrections to separate formulations for viscoelastic effects from computations for incompressible fluids, and our main contribution lies in separating the formulations not to slow down the computations for fluid incompressibility. To show our idea is applicable to different geometrically-motivated methods and to compare the existing methods with ours under fair conditions, we offer three schemes for correcting velocities of particles without estimating viscoelastic force fields for fast and stable simulations. The first scheme adopts SM [27] to derive suitable transformations of particle sets. The second is based on Hooke's law to compute attraction forces. The last utilizes the idea of *Position-Based Dynamics* [26] (PBD) to compute velocity correction vectors. We employ IISPH [17] as our underlying fluid solver owing to the fast convergence of density fluctuations for the incompressible flow, and combine IISPH [17] with one of the three velocity correction schemes to achieve volume preserving viscoelastic fluids. Fig. 1 illustrates horizontally spreading motions of a viscoelastic ball compressed by a moving solid, which cannot be fully captured with compressible viscoelastic fluids.

2 Related Work

Viscoelastic fluids Viscoelastic fluids have mainly been simulated using a spring-based method or methods based on SPH. Miller and Pearce [22] and Terzopoulos et al. [39] proposed a spring-based model that computes repulsion and attraction forces among particles to simulate viscoelastic materials. Their method was also adopted by Steele et al. [34] and Tamura et al. [38]. Clavet et al. [10] combined this spring-based method with SPH to simulate materials with elasticity, plasticity, and viscosity, adopting a prediction-relaxation scheme, which partly shares the features of PBD. Takamatsu and Kanai [37] fast and robustly simulated viscoelastic materials by combining SPH with a variant of SM. Müller et al. [28] proposed an elastic term which uses *Moving Least Square* (MLS) to simulate elastoplastic objects. Solenthaler et al. [33] adopted the formulation of this elastic term and computed it using SPH instead of MLS to allow for robustly simulating fluid with various properties under the condition of collinear or coplanar particle distributions. The method of Solenthaler et al. [33] was extended to handle rotational motions of elastic materials [3]. Mao et al. [21] introduced an elastic force term, called nonlinear corotational Maxwell model, into the Navier-Stokes equations to simulate viscoelastic fluids. This method was also adopted by Chang et al. [8]. Paiva et al. [29] simulated viscoplastic objects using a generalized Newtonian model.

Gerszewski et al. [13] proposed a new formulation for simulating elastoplastic materials, which uses affine transformations to compute the gradient of deformations. This formulation was solved by Zhou et al. [43] in an implicit manner to efficiently and robustly perform simulations adopting larger time steps.

Takahashi and Fujishiro [35] simulated viscous fluids with elasticity using PBD with a low computational cost.

Volume preservation Volume preservation is necessary to plausibly generate motions of objects, and many methods for preserving volumes have been proposed for deformable objects by computing deformation energy [40,30] or imposing constraints upon computational elements [16,19,36,12].

For particle-based fluids, volume preservation methods have focused on minimizing density fluctuations from the fluid rest density by enforcing the incompressibility of fluid [18], because of frequent topology changes unlike methods for deformable objects. After Müller et

al. [25] presented the basic SPH method, which suffers from unacceptable density fluctuations and volume loss due to the dependence on an equation of state (EOS), Becker and Teschner [4] proposed *Weakly Compressible SPH* and alleviated the fluctuations by replacing the EOS in [25] with a stiffer EOS called Tait equation [24], necessitating excessively smaller time steps. In order to adopt larger ones, predictive-correction schemes were proposed, e.g., PCISPH [32] and local Poisson SPH [15]. Recently, Bodin et al. [6] proposed a system of velocity constraints to improve the accuracy of PCISPH achieving uniform density fields over particles. To further accelerate fluid simulations using PCISPH, Macklin and Müller [20] adapted PBD for fluid simulation to allow for the use of larger time steps than the ones adopted in PCISPH. Similarly, Ihmsen et al. [17] proposed IISPH, which computes particle pressure values in an implicit manner with minimal assumptions. Furthermore, Cornelis et al. [11] combined IISPH with *Fluid-Implicit-Particle* to further accelerate simulation performances.

3 Our Algorithm

Our key idea to simulate volume preserving viscoelastic fluids is to separately deal with the volume preservation and viscoelasticity. While preserving the fluid volumes by enforcing the incompressibility of fluid using IISPH [17], we utilize velocity corrections for viscoelastic effects without negatively influencing the convergence of density fluctuations in IISPH [17].

In Section 3.1, we first make a brief explanation of IISPH [17] to elucidate the mechanism of the fluid solver and its important features, and then Section 3.2 shows the procedure of our method. The three velocity correction schemes are detailed in Section 4.

3.1 IISPH Method

IISPH aims to achieve deviations of the particle density ρ less than a certain percentage, e.g. 1%, to the fluid rest density ρ_0 by iteratively correcting the pressure of each particle in an implicit manner. For the derivation of formulations and implementation details, please refer to the original paper of IISPH [17].

First, we compute the particle density $\rho_i = \sum_j m_j W_{ij}$ for particle i with its mass m_i, neighboring particle j, and finite support kernel $W_{ij} = W(\mathbf{x}_{ij}, h)$, where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, $\mathbf{x}_i$ is the position of particle i, and h denotes a kernel radius. Then, intermediate density $\tilde{\rho}_i$ is computed with intermediate velocity $\tilde{\mathbf{v}}_i = \mathbf{v}_i + \Delta t \tilde{\mathbf{F}}_i / m_i$, where $\mathbf{v}_i$ is the velocity of particle i and $\tilde{\mathbf{F}}_i$ is non-pressure force, using the continuity equation by

$$\tilde{\rho}_i = \rho_i + \Delta t \sum_j m_j \tilde{\mathbf{v}}_{ij} \nabla W_{ij},$$

where Δt is a time step, and $\tilde{\mathbf{v}}_{ij} = \tilde{\mathbf{v}}_i - \tilde{\mathbf{v}}_j$. The next step is to find pressure values which result in appropriate pressure forces to make all particles satisfy the rest density ρ_0, and we obtain the following equations:

$$\begin{aligned}
\rho_0 - \tilde{\rho}_i &= p_i a_{ii} + b_{ij}, \\
a_{ii} &= \sum_j m_j (\mathbf{d}_{ii} - \mathbf{d}_{ji}) \cdot \nabla W_{ij}, \\
b_{ij} &= \sum_j m_j (\sum_j \mathbf{d}_{ij} p_j - \mathbf{d}_{jj} p_j - \sum_{k \neq i} \mathbf{d}_{jk} p_k) \cdot \nabla W_{ij}, \\
\mathbf{d}_{ii} &= -\Delta t^2 \sum_j \frac{m_j}{\rho_i^2} \nabla W_{ij}, \\
\mathbf{d}_{ij} &= -\Delta t^2 \frac{m_j}{\rho_j^2} \nabla W_{ij}.
\end{aligned}$$

Then, we compute pressure values using weighted Jacobi method with an iteration index n and a relaxation factor w:

$$p_i^{n+1} = (1-w) p_i^n + w \frac{\rho_0 - \tilde{\rho}_i - b_{ij}^n}{a_{ii}}.$$

Note that the computations of the weighted Jacobi method include particle positions, and thus the positions must be kept during the iterations after intermediate velocity $\tilde{\mathbf{v}}$ is determined. As explained in [32] and [17], ignoring position changes of particles is the cause of erroneous sets of neighboring particles and their inter-particle distances, and leads to inaccurate estimation of physical values, which can cause the fluid solver to fail or slow down the convergence of the iterations.

After we obtain pressure p_i, pressure force $\mathbf{F}_i^p$ is computed by

$$\mathbf{F}_i^p = -m_i \sum_j m_j (\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2}) \nabla W_{ij}, \tag{1}$$

and particle velocity $\mathbf{v}_i$ and position $\mathbf{x}_i$ are integrated using the Euler-Cromer scheme.

Since we get pressure p from intermediate velocity $\tilde{\mathbf{v}}$, IISPH is interpreted as a function that $p = IISPH(\tilde{\mathbf{v}})$, and this corresponds to the pressure projection used in the Eulerian fluid simulation [7].

3.2 Procedure of Our Method

We insert the velocity correction step (Section 4) before the iterations in IISPH, and the following is our simulation algorithm.

1. Find neighboring particles;

2. Correct particle velocity (see Section 4);
   (a) Compute predicted velocity $\mathbf{v}^{\mathrm{adv}}$;
   (b) Get corrected velocity $\mathbf{v}^*$ using one of the three schemes;
   (c) Obtain intermediate velocity $\tilde{\mathbf{v}}$ using XSPH [23];
3. Compute pressure value $p = IISPH(\tilde{\mathbf{v}})$;
4. Compute pressure force $\mathbf{F}^p$ using Eq. (1);
5. Update velocity $\mathbf{v} = \tilde{\mathbf{v}} + \Delta t \mathbf{F}^p / m$;
6. Update position $\mathbf{x} = \mathbf{x} + \Delta t \mathbf{v}$.

4 Velocity Correction

We correct particle velocities to describe viscoelastic effects without changing particle positions. In order to obtain intermediate velocity $\tilde{\mathbf{v}}_i$ for particle i as an input for IISPH, we first compute predicted velocity $\mathbf{v}_i^{\mathrm{adv}}$ with all forces $\mathbf{F}_i^{\mathrm{adv}}$ except for viscoelastic and pressure ones without directly computing $\tilde{\mathbf{v}}_i$ as in IISPH:

$$\mathbf{v}_i^{\mathrm{adv}} = \mathbf{v}_i + \Delta t \frac{\mathbf{F}_i^{\mathrm{adv}}}{m_i}.$$

Then, we correct particle velocities with velocity correction vector $\Delta \mathbf{v}_i$:

$$\mathbf{v}_i^* = \mathbf{v}_i^{\mathrm{adv}} + \Delta \mathbf{v}_i.$$

Lastly, we apply XSPH [23] to reduce particle oscillations with $\mathbf{v}_{ij}^* = \mathbf{v}_i^* - \mathbf{v}_j^*$ and a velocity mixing parameter ϵ $(0 \leq \epsilon \leq 1)$:

$$\tilde{\mathbf{v}}_i = \mathbf{v}_i^* + \epsilon \sum_j \frac{m_j}{\rho_j} \mathbf{v}_{ji}^* W_{ij}.$$

Note that XSPH [23] modifies only particle velocities without affecting the convergence of the fluid solver, and therefore XSPH [23] is suitable for the purpose of our velocity correction.

In the followings, we explain the three schemes for computing velocity correction vector $\Delta \mathbf{v}$.

4.1 Shape Matching Scheme

We formulate shape matching scheme as with the method of [37] adapting *Lattice Shape Matching* [31], which was extended from the original SM [27] to increase the degree of freedom of object deformations.

We consider particle i with its reference position $\mathbf{x}_i^0$, which has a set of S_i^{sm} particles for SM, and the reference position $\mathbf{s}_i^0$ of the set is defined to be the center of mass of the particles. In SM, particle i is pulled toward its goal position $\mathbf{g}_i$ to restore the original configuration of the particles, and individual goal positions are computed to match the original configuration of the particles defined by $\mathbf{x}_i^0$ with current particle distributions denoted as $\mathbf{x}_i$ after the particles are transferred. Specifically, we compute a rotational matrix $\mathbf{R}_i$ which transforms the particles to their own reference positions with $\mathbf{s}_i$, the current center of mass of the particles. The goal position $\mathbf{g}_i$ is computed by

$$\begin{aligned}\mathbf{g}_i &= \frac{1}{S_i^{\mathrm{sm}}} \sum_j^{S_i^{\mathrm{sm}}} \left(\mathbf{R}_j (\mathbf{x}_i^0 - \mathbf{x}_j^0) + \mathbf{s}_j \right) \\ &= \mathbf{R}_i (\mathbf{x}_i^0 - \mathbf{s}_i^0) + \mathbf{s}_i,\end{aligned}$$

and then we obtain velocity correction vector $\Delta \mathbf{v}_i^{\mathrm{sm}}$ with a coefficient for the shape matching velocity correction k_i^{sm} $(0 \leq l_i^{\mathrm{sm}} \leq k_i^{\mathrm{sm}} \leq 1)$ and l_i^{sm}, the lower limit of k_i^{sm}:

$$\Delta \mathbf{v}_i^{\mathrm{sm}} = k_i^{\mathrm{sm}} \frac{\mathbf{g}_i - \mathbf{x}_i}{\Delta t},$$

If $l_i^{\mathrm{sm}} < k_i^{\mathrm{sm}}$, and $\alpha_i^{\mathrm{sm}} < (||\mathbf{g}_i - \mathbf{x}_i||)/h$, where α_i^{sm} $(0 \leq \alpha_i^{\mathrm{sm}})$ is a yield criterion, which invalidates the influence of small oscillations and deformations of materials, we weaken the effect of k_i^{sm} to realize plastic and viscous effects by

$$k_i^{\mathrm{sm}} \leftarrow \max(k_i^{\mathrm{sm}} - \Delta t k_i^{\mathrm{sm}} d_i^{\mathrm{sm}}, l_i^{\mathrm{sm}}),$$

where d_i^{sm} $(0 \leq d_i^{\mathrm{sm}})$ is a parameter that controls the weakening speed of k_i^{sm}.

In order to enable the merge of viscoelastic materials, we change a set of particles for SM. Let c_{ij}^{sm} denote the distance of two particles which are not associated with each other as a particle for SM, and we update the set if $c_{ij}^{\mathrm{sm}}/h < \beta_{ij}^{\mathrm{sm}}$, where $\beta_{ij}^{\mathrm{sm}} = (\beta_i^{\mathrm{sm}} + \beta_j^{\mathrm{sm}})/2$, $(0 < \beta_i^{\mathrm{sm}})$, is a threshold to update a set of particles for SM. We also update the set for the split of the materials if $\gamma_i^{\mathrm{sm}} < ||\mathbf{g}_i - \mathbf{x}_i||/h$, where $\gamma_i^{\mathrm{sm}} (0 < \gamma_i^{\mathrm{sm}})$ is a parameter which limits the extent of deformations.

4.2 Spring-based Scheme

In the spring-based scheme, we correct particle velocities with springs between two particles, and the springs are created when an object is generated, Hereafter we call particles connected with their springs as *connected* particles.

Since IISPH that we use can resolve fluid compression, we address only fluid expansions which are caused by separating particles from the other. In order to compute velocity correction vector $\mathbf{v}_i^{\mathrm{spr}}$ only with connected particles which are separating from the other, we define a distance function with their positions and interparticle distance r_{ij} as

$$D(\mathbf{x}_i, \mathbf{x}_j) = \max(||\mathbf{x}_{ij}|| - r_{ij}, 0), \tag{2}$$

where r_{ij} is initialized with the distance between the two particles when their spring is created. Then, we

compute velocity correction vector $\mathbf{v}_i^{\mathrm{spr}}$ based on Hooke's law with k_i^{spr} $(0 \leq l_i^{\mathrm{spr}} \leq k_i^{\mathrm{spr}})$, a coefficient for the spring-based velocity correction and l_i^{spr}, the lower limit of k_i^{spr} by

$$\Delta \mathbf{v}_i^{spr} = -\frac{\Delta t}{m_i} \sum_j^{S_i^{\mathrm{spr}}} \frac{k_i^{\mathrm{spr}} + k_j^{\mathrm{spr}}}{2} D(\mathbf{x}_i, \mathbf{x}_j) \frac{\mathbf{x}_{ij}}{||\mathbf{x}_{ij}||},$$

where S_i^{spr} denotes the number of connected particles to particle i. Similar to the shape matching scheme, we weaken the effect of k_i^{spr} if $l_i^{\mathrm{spr}} < k_i^{\mathrm{spr}}$:

$$k_i^{\mathrm{spr}} \leftarrow \max(k_i^{\mathrm{spr}} - \frac{\Delta t}{S_i^{\mathrm{spr}}} \sum_j^{S_i^{\mathrm{spr}}} d_{ij}^{\mathrm{spr}} \sigma_{ij}, l_i^{\mathrm{spr}}),$$

$$\sigma_{ij} = \begin{cases} 1 & \text{if } \alpha_{ij}^{\mathrm{spr}} < \frac{D(\mathbf{x}_i, \mathbf{x}_j)}{h}, \\ 0 & \text{otherwise,} \end{cases}$$

where $d_{ij}^{\mathrm{spr}} = (d_i^{\mathrm{spr}} + d_j^{\mathrm{spr}})/2$, $(0 \leq d_i^{\mathrm{spr}})$, governs the weakening speed of k_i^{spr}, and $\alpha_{ij}^{\mathrm{spr}} = (\alpha_i^{\mathrm{spr}} + \alpha_j^{\mathrm{spr}})/2$, $(0 \leq \alpha_i^{\mathrm{spr}})$, is a yield criterion.

In the spring-based scheme, we also enable the merge and split of viscoelastic materials. Let c_{ij}^{spr} denote the distance of two particles which are not interconnected, and we generate a new spring between the two particles if $c_{ij}^{\mathrm{spr}}/h < \beta_{ij}^{\mathrm{spr}}$, where $\beta_{ij}^{\mathrm{spr}} = (\beta_i^{\mathrm{spr}} + \beta_j^{\mathrm{spr}})/2$, $(0 < \beta_i^{\mathrm{spr}})$, is a threshold to generate springs. By contrast, we remove the spring between two particles if $\gamma_{ij}^{\mathrm{spr}} < ||\mathbf{x}_{ij} - r_{ij}||/h$, where $\gamma_{ij}^{\mathrm{spr}} = (\gamma_i^{\mathrm{spr}} + \gamma_j^{\mathrm{spr}})/2$, $(0 < \gamma_i^{\mathrm{spr}})$, is a parameter which restricts the extent of deformations.

4.3 Position-based Scheme

The position-based scheme is similar to the spring-based one since particle velocities are corrected based on pairwise connections between particles, and the connections are added and removed as in the spring-based scheme. The differences between the two schemes lie in how to correct particle velocities and select relevant parameters. This position-based scheme relies on positional differences of particles to compute velocity correction vector $\Delta \mathbf{v}_i^{\mathrm{pb}}$ by adapting the idea of PBD [26,5] instead of computing attraction forces:

$$\Delta \mathbf{v}_i^{\mathrm{pb}} = -\frac{1}{\Delta t} \sum_j^{S_i^{\mathrm{pb}}} \frac{k_i^{\mathrm{pb}} + k_j^{\mathrm{pb}}}{2} \frac{m_j}{m_i + m_j} D(\mathbf{x}_i, \mathbf{x}_j) \frac{\mathbf{x}_{ij}}{||\mathbf{x}_{ij}||},$$

where S_i^{pb} is the number of connected particles to particle i, k_i^{pb} $(0 \leq l_i^{\mathrm{pb}} \leq k_i^{\mathrm{pb}} \leq 1)$ is a coefficient for the position-based velocity correction, and l_i^{pb}, the lower limit of k_i^{pb}. Since the range of k_i^{pb} is bounded by two sides, tuning parameters can be easier in the position-based scheme than the spring-based one. We correct particle velocities and modify k^{pb} as in the spring-based scheme.

Table 1 Simulation conditions and performances.

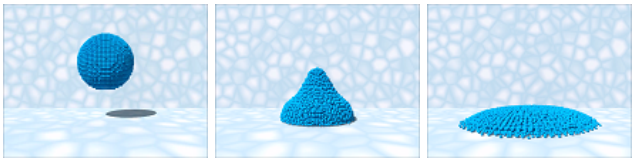
Fig.	# of particles fluid/solid	Total time	Total simulation steps
1	65.8k/45.0k	10m:16s	60
2 (a)	17.1k/23.3k	5m:53s	150
2 (b)	17.1k/23.3k	4m:51s	150
2 (c)	17.1k/23.3k	4m:55s	150
3 (a)	8.2k/13.4k	6m:28s	200
3 (b)	8.2k/13.4k	6m:46s	200
5 (a)	8.2k/13.4k	6m:58s	150
5 (b)	8.2k/13.4k	6m:31s	150
7	33.4k/40.7k	12m:15s	200
8	45.2k/36.4k	58m:05s	170

5 Results

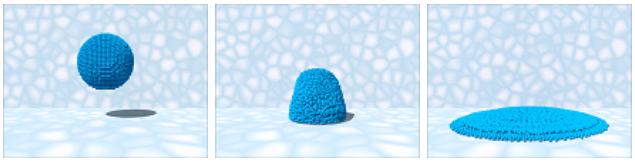
We implemented our method in C++ and parallelized it with OpenMP 2.0. All the simulations were performed on a PC with a 4-core Intel Core i7 3.50 GHz CPU and RAM 16.0 GB. We gained accelerated simulation performances by a factor from three to five with six threads in total. In our simulations, fluid-solid coupling is achieved by the method of Akinci et al. [1]. For fast simulations, we used only 2 iterations for computing pressure values in IISPH except Figs. 1, 3, and 5. In all the simulations, parameters were empirically adjusted. Our velocity correction schemes occupy roughly 16.4% of the total computational time with 2 pressure iteration in IISPH. We used off-line renderer POV-Ray 3.7 and the rendering needed about 20 seconds per frame on average for Fig. 1. The accompanying video includes several results produced with our method or existing methods. We tabulate the simulation conditions and performances in Table 1.

5.1 Velocity Correction Schemes

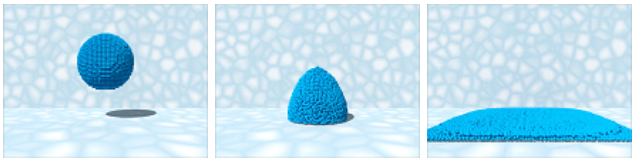
We compare the motions of viscoelastic fluids produced with the three different schemes for velocity corrections in order to clarify how the selection of the schemes affects the resulting behaviors. Fig. 2 displays the motion of a viscoelastic ball dropped onto the ground, where we used the same number of neighboring particles in all the schemes, and we can plausibly generate characteristic behaviors of viscoelastic fluids. Our method using the shape matching scheme (Fig. 2 (a)) enables smooth transfers of deformation waves, which leads to preserving energy for a longer time as compared to the others. Since the three schemes are not physically-based, we



(a) Shape matching scheme

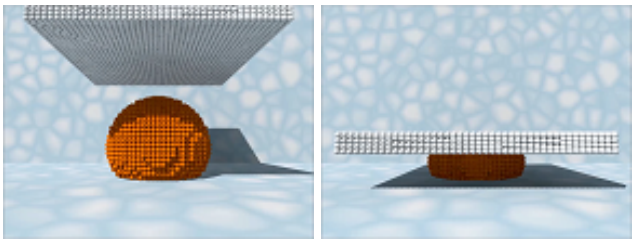


(b) Spring-based scheme

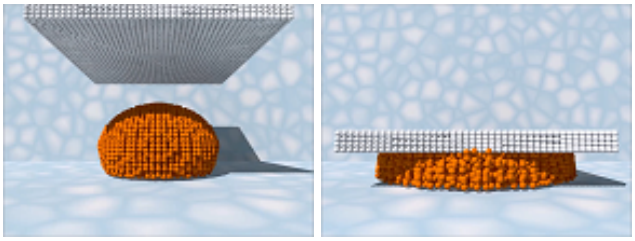


(c) Position-based scheme

Fig. 2 Motion comparisons of the three different velocity correction schemes.



(a) Method of Takamatsu and Kanai [37]



(b) Our method using the shape matching scheme

Fig. 3 A volume comparison with a viscoelastic sphere compressed by a moving solid along a fixed path.

need to carefully adjust simulation parameters to generate similar results with different schemes.

As shown in Table 1, the simulation costs of the methods using the spring-based scheme (Fig. 2 (b)) and the position-based one (Fig. 2 (c)) are almost the same. However, the method using the shape matching scheme can be a bit costly due to the singular value decomposition in SM, and besides needs more particles in the set for SM to stabilize the simulation than the others.

5.2 Volume Preservation

We compare the existing methods and ours through interactions between a viscoelastic ball with a moving solid along a fixed path.

Shape matching scheme First, we elucidate the difference between the method of Takamatsu and Kanai [37], who used the combination of SPH and SM, and ours using the shape matching scheme for velocity corrections. Although Takamatsu and Kanai [37] employed the basic SPH as their underlying fluid solver, which suffers from significant volume loss, we replace the SPH with IISPH for a fair comparison. Namely, we here regard the method in [37] as the combination of IISPH and SM. In this scene, we used 5 pressure iterations in IISPH, and we estimated an occupied volume V by a summation of particles' volume as $V = \sum_i V_i = \sum_i m_i/\hat{\rho}_i$, where V_i is the volume of particle i, and $\hat{\rho}_i$ the estimated density after only the particle i is moved using Laplacian smoothing to reduce the effect of the underestimated particle density on fluid surfaces.

Fig. 3 compares the two methods, and Fig. 4 shows the profile of the comparison in terms of the volume V. With the method of Takamatsu and Kanai [37], the volume of the ball fluctuates due to the first collision with the ground, and decreases and increases according to the movement of the solid. In contrast, our method can preserve the volume of the ball, which is fairly close to the reference value, and achieve less fluctuations in the volume. Additionally, the ball exhibits horizontally spreading motions when being compressed, which cannot be captured fully with the method in [37].

Spring-based scheme Second, we compare the method of Clavet et al. [10], who combined SPH with a spring-based method adopting a prediction-relaxation scheme, with ours using the spring-based scheme. As with the above previous method, since Clavet et al. [10] used the basic SPH, we again use IISPH as the underlying fluid solver of their method. In this scenario, we set the volume loss of 0.01% as a convergence criteria in IISPH to clarify the differences of the two methods.

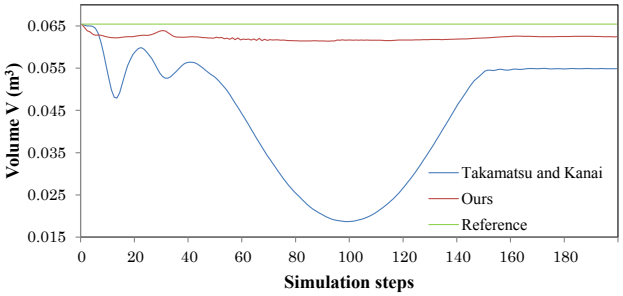
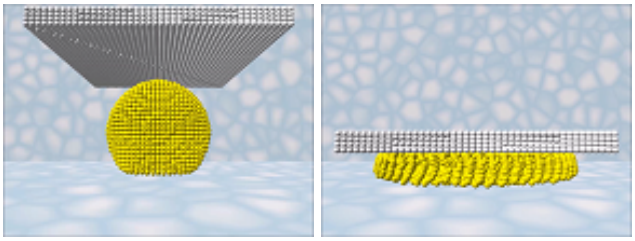
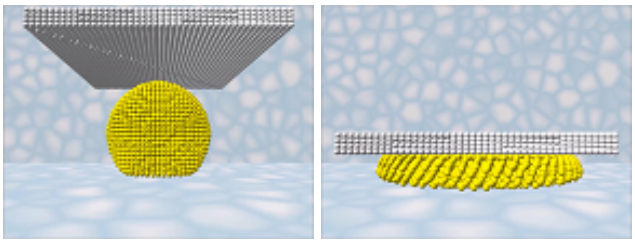


Fig. 4 Changes in the volume of the viscoelastic sphere in **Fig. 3** according to simulation steps.



(a) Method of Clavet et al. [10]



(b) Our method using the spring-based scheme

Fig. 5 A comparison with a viscoelastic sphere compressed by a moving solid along a fixed path.

Fig. 5 compares the two methods, and Fig. 6 gives the profile of the number of the needed iterations in Fig. 5. The method of Clavet et al. [10] changes particle positions in the prediction-relaxation scheme, and these changes make the fluid solver fail or slow down convergence of the iterations. As illustrated in Fig. 6, the method of Clavet et al. [10] requires more pressure iterations to enforce the incompressibility of the material than our method when the material is compressed by the solid, because spring forces tend to be strong and lead to significant position displacements. Additionally, though IISPH can produce satisfactory pressure values to achieve the incompressibility of fluid, resulting positions through the time integration are not optimal, and hence can cause significant oscillations in [10].

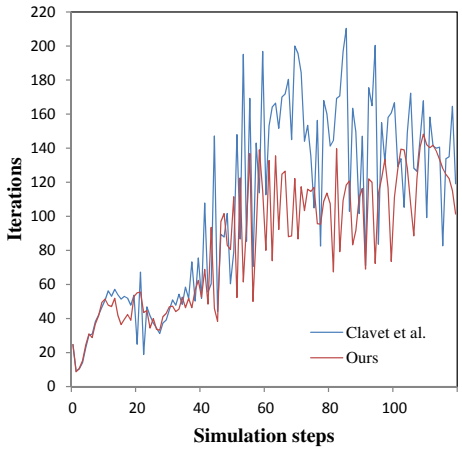


Fig. 6 A profile of the number of needed pressure iterations to achieve the volume loss of the material less than 0.01% in **Fig. 5** according to simulation steps.

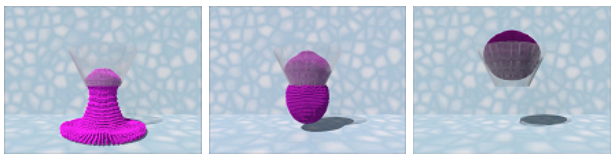


Fig. 7 A ball with different viscoelasticity values dropped onto a funnel (low viscoelascitiy: light purple, and high one: dark purple).

5.3 Different Viscoelasticity Values

Fig. 7 demonstrates the effect of different viscoelasticity values. In this scene, a ball is dropped onto a funnel. While the balls with lower viscoelasticity values smoothly fall through the funnel, the highly viscoelastic ball needs longer time to pass through the funnel. Additionally, in Figs. 7 (b) and (c), we can observe the Barus effect (also known as die swell, exclude swell, and Merrington effect) that a stream of viscoelastic fluids swells wider than the diameter of an opening when the stream goes through the opening. Our velocity correction method can plausibly generate such a special effect of viscoelastic fluids.

Fig. 8 illustrates the merge of the two balls with different viscoelasticity values. Due to the update of particle connections, we can easily and naturally generate adhesive behavior of the two viscoelastic balls.

6 Conclusions

We have presented a new particle-based method for simulating volume preserving viscoelastic fluids by correcting particle velocities based on particle positions. The volume preservation is achieved by separating for-

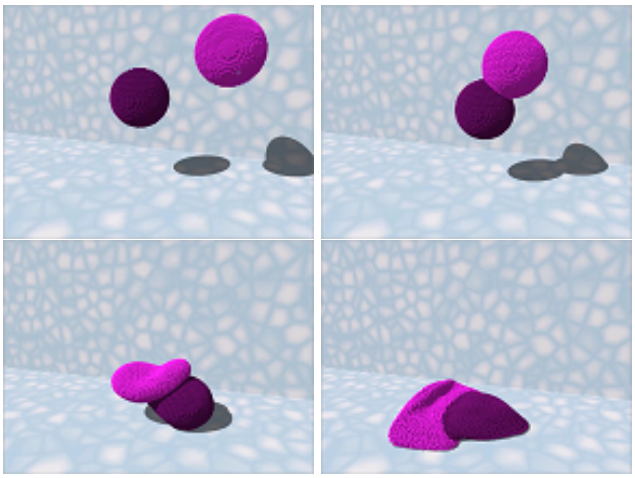


Fig. 8 Two balls with different viscoelasticity values colliding with each other (low viscoelasticity: light purple and high one: dark purple).

mulations for viscoelastic effects from computations for incompressible fluids. To correct particle velocities, we have provided three different schemes: shape matching, spring-based, and position-based velocity corrections. We have empirically proven that the three schemes can plausibly simulate viscoelastic fluids, and that our method can preserve fluid volumes generating characteristic viscoelastic motions such as Barus effect.

Since our method utilizes the geometric relations of particles to correct particle velocities with several simulation parameters instead of estimating viscoelastic force fields, we cannot adopt real physical properties, and needs to empirically adjust the parameters through experiments in order to generate desirable viscoelastic effects. For future work, we plan to address the problem by the control of fluid behavior, and investigating further how to find appropriate simulation parameters.

References

1. Akinci, N., Ihmsen, M., Akinci, G., Solenthaler, B., Teschner, M.: Versatile rigid-fluid coupling for incompressible SPH. ACM Transactions on Graphics **31**(4), 1–8 (2012)
2. Bargteil, A.W., Wojtan, C., Hodgins, J.K., Turk, G.: A finite element method for animating large viscoplastic flow. ACM Transactions on Graphics **26**(3) (2007)
3. Becker, M., Ihmsen, M., Teschner, M.: Corotated SPH for deformable solids. In: Proceedings of the Fifth Eurographics Conference on Natural Phenomena, pp. 27–34 (2009)
4. Becker, M., Teschner, M.: Weakly compressible SPH for free surface flows. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 209–217 (2007)
5. Bender, J., Müller, M., Otaduy, M.A., Teschner, M.: Position-based methods for the simulation of solid objects in computer graphics. In: EUROGRAPHICS 2013 State of the Art Reports, pp. 1–22 (2013)
6. Bodin, K., Lacoursiere, C., Servin, M.: Constraint fluids. IEEE Transactions on Visualization and Computer Graphics **18**(3), 516–526 (2012)
7. Bridson, R.: Fluid simulation for computer graphics. A K Peters / CRC Press (2008)
8. 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, pp. 111–117 (2009)
9. Clausen, P., Wicke, M., Shewchuk, J.R., O'Brien, J.F.: Simulating liquids and solid-liquid interactions with lagrangian meshes. ACM Transactions on Graphics **32**(2), 1–15 (2013)
10. Clavet, S., Beaudoin, P., Poulin, P.: Particle-based viscoelastic fluid simulation. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 219–228 (2005)
11. Cornelis, J., Ihmsen, M., Peer, A., Teschner, M.: IISPH-FLIP for incompressible fluids. Computer Graphics Forum **33**(2), 255–262 (2014)
12. Diziol, R., Bender, J., Bayer, D.: Robust real-time deformation of incompressible surface meshes. In: Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 237–246 (2011)
13. Gerszewski, D., Bhattacharya, H., Bargteil, A.W.: A point-based method for animating elastoplastic solids. In: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 133–138 (2009)
14. Goktekin, T.G., Bargteil, A.W., O'Brien, J.F.: A method for animating viscoelastic fluids. ACM Transactions on Graphics **23**(3), 463–468 (2004)
15. He, X., Liu, N., Li, S., Wang, H., Wang, G.: Local poisson SPH for viscous incompressible fluids. Computer Graphics Forum **31**(6), 1948–1958 (2012)
16. Hong, M., Jung, S., Choi, M.H., Welch, S.W.: Fast volume preservation for a mass-spring system. IEEE Computer Graphics and Applications **26**(5), 83–91 (2006)
17. Ihmsen, M., Cornelis, J., Solenthaler, B., Horvath, C., Teschner, M.: Implicit incompressible sph. IEEE Transactions on Visualization and Computer Graphics **20**(3), 426–435 (2014)
18. Ihmsen, M., Orthmann, J., Solenthaler, B., Kolb, A., Teschner, M.: SPH fluids in computer graphics. In: EUROGRAPHICS 2014 State of the Art Reports, pp. 21–42 (2014)
19. Irving, G., Schroeder, C., Fedkiw, R.: Volume conserving finite element simulations of deformable models. ACM Transactions on Graphics **26**(3) (2007)
20. Macklin, M., Müller, M.: Position based fluids. ACM Transactions on Graphics **32**(4), 1–5 (2013)
21. Mao, H., Yang, Y.H.: Particle-based non-newtonian fluid animation with heating effects. Tech. rep., University of Alberta (2006)
22. Miller, G., Pearce, A.: Globular dynamics: A connected particle system for animating viscous fluids. Computers and Graphics **13**(3), 305–309 (1989)
23. Monaghan, J.J.: On the problem of penetration in particle methods. Journal of Computational Physics **82**(1), 1–15 (1989)
24. Monaghan, J.J.: Simulating free surface flows with SPH. Journal of Computational Physics **110**(2), 399–406 (1994)
25. 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, pp. 154–159 (2003)

26. Müller, M., Heidelberger, B., Hennix, M., Ratcliff, J.: Position based dynamics. Journal of Visual Communication and Image Representation **18**(2), 109–118 (2007)
27. Müller, M., Heidelberger, B., Teschner, M., Gross, M.: Meshless deformations based on shape matching. ACM Transactions on Graphics **24**(3), 471–478 (2005)
28. Müller, M., Keiser, R., Nealen, A., Pauly, M., Gross, M., Alexa, M.: Point based animation of elastic, plastic and melting objects. In: Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 141–151 (2004)
29. Paiva, A., Petronetto, F., Lewiner, T., Tavares, G.: Particle-based viscoplastic fluid/solid simulation. Computer-Aided Design **41**(4), 306–314 (2009)
30. Patterson, T., Mitchell, N., Sifakis, E.: Simulation of complex nonlinear elastic bodies using lattice deformers. ACM Transactions on Graphics **31**(6), 197:1–197:10 (2012)
31. Rivers, A.R., James, D.L.: FastLSM: Fast lattice shape matching for robust real-time deformation. ACM Transactions on Graphics **26**(3) (2007)
32. Solenthaler, B., Pajarola, R.: Predictive-corrective incompressible SPH. ACM Transactions on Graphics **28**(3), 1–6 (2009)
33. Solenthaler, B., Schläfli, J., Pajarola, R.: A unified particle model for fluid solid interactions. Computer Animation and Virtual Worlds **18**(1), 69–82 (2007)
34. Steele, K., Cline, D., Egbert, P.K., Dinerstein, J.: Modeling and rendering viscous liquids. Computer Animation and Virtual Worlds **15**(3-4), 183–192 (2004)
35. Takahashi, T., Fujishiro, I.: Accelerated viscous fluid simulation using position-based constraints. In: Proceedings of CAD/Graphics 2013, pp. 260–267 (2013)
36. Takamatsu, K., Kanai, T.: Volume-preserving LSM deformations. In: ACM SIGGRAPH ASIA 2009 Sketches, pp. 15:1–15:1 (2009)
37. Takamatsu, K., Kanai, T.: A fast and practical method for animating particle-based viscoelastic fluids. The International Journal of Virtual Reality **10**, 29–35 (2011)
38. Tamura, N., Nakaguchi, T., Tsumura, N., Miyake, Y.: Spring-bead animation of viscoelastic materials. IEEE Computer Graphics and Applications **27**(6), 87–93 (2007)
39. Terzopoulos, D., Platt, J., Fleischer, K.: Heating and melting deformable models. Journal of Visualization and Computer Animation **2**, 68–73 (1991)
40. Teschner, M., Heidelberger, B., Muller, M., Gross, M.: A versatile and robust model for geometrically complex deformable solids. In: Proceedings of the Computer Graphics International, pp. 312–319 (2004)
41. Wicke, M., Ritchie, D., Klingner, B.M., Burke, S., Shewchuk, J.R., O'Brien, J.F.: Dynamic local remeshing for elastoplastic simulation. ACM Transactions on Graphics **29**, 1–11 (2010)
42. Wojtan, C., Turk, G.: Fast viscoelastic behavior with thin features. ACM Transactions on Graphics **27**(3), 1–8 (2008)
43. Zhou, Y., Lun, Z., Kalogerakis, E., Wang, R.: Implicit integration for particle-based simulation of elasto-plastic solids. Computer Graphics Forum **32**(7), 215–223 (2013)