An Implicit Viscosity Formulation for SPH Fluids

Andreas Peer* Markus Ihmsen Jens Cornelis Matthias Teschner
University of Freiburg

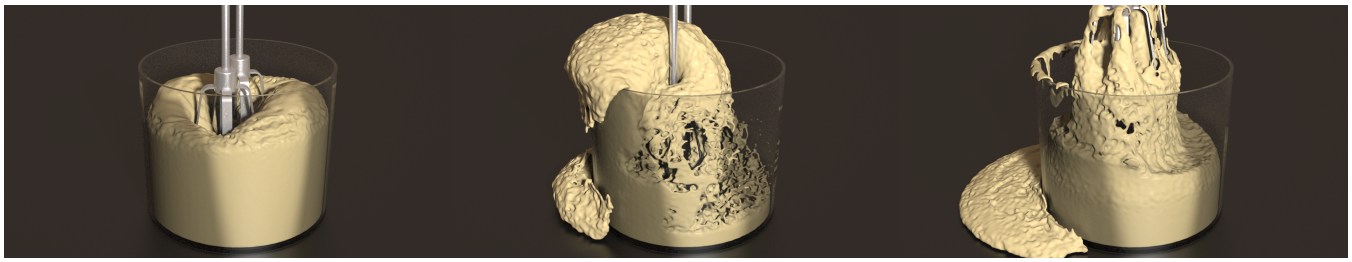


Figure 1: *Interaction of a highly viscous fluid with complex moving solids. Up to 780k particles are used. The computation time is 30s per frame.*

Abstract

We present a novel implicit formulation for highly viscous fluids simulated with Smoothed Particle Hydrodynamics SPH. Compared to explicit methods, our formulation is significantly more efficient and handles a larger range of viscosities. Differing from existing implicit formulations, our approach reconstructs the velocity field from a target velocity gradient. This gradient encodes a desired shear-rate damping and preserves the velocity divergence that is introduced by the SPH pressure solver to counteract density deviations. The target gradient ensures that pressure and viscosity computation do not interfere. Therefore, only one pressure projection step is required, which is in contrast to state-of-the-art implicit Eulerian formulations. While our model differs from true viscosity in that vorticity diffusion is not encoded in the target gradient, it nevertheless captures many of the qualitative behaviors of viscous liquids. Our formulation can easily be incorporated into complex scenarios with one- and two-way coupled solids and multiple fluid phases with different densities and viscosities.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: Physically-based animation, fluid simulation, Smoothed Particle Hydrodynamics, viscosity

1 Introduction

Simulating highly viscous fluids such as honey, mud, toothpaste or dough as shown in Fig. 1 is involved. Explicit approaches require small timesteps, e.g., [Monaghan 1989; Foster and Metaxas 1996; Morris et al. 1997], while implicit formulations need to solve a linear system, e.g., [Stam 1999; Carlson et al. 2002; Rasmussen et al. 2004; Batty and Bridson 2008]. As implicit formulations work with significantly larger timesteps, they are commonly preferred over explicit formulations.

In highly viscous fluids, viscosity and incompressibility constraints can interfere. This is particularly true for bulk viscosity that influences the divergence of the velocity field. In explicit formulations with small timesteps, competing constraints might be neglected. In implicit formulations with large timesteps, however, this issue has to be addressed. In this context, viscosity formulations for incompressible Eulerian fluids commonly assume that the input velocity field is divergence-free. As the resulting velocity field is not necessarily divergence-free, two pressure projection steps are performed, e.g. [Losasso et al. 2006; Batty and Bridson 2008].

Contribution: This paper proposes a novel implicit formulation for highly viscous SPH fluids that particularly addresses the interference of pressure and viscosity computation. Therefore, a target velocity gradient is employed that does not only encode the desired viscosity, but also preserves arbitrary velocity divergences that might have been introduced by an SPH pressure solver. In contrast to incompressible Eulerian techniques, SPH pressure solvers, e.g. implicit incompressible SPH (IISPH) [Ihmsen et al. 2014a; Ihmsen et al. 2014b], commonly introduce some divergence to the velocity field to counteract density deviations. Our formulation explicitly preserves density corrections that have been computed in the preceding pressure projection. This also means that only one pressure projection step is required, which is in contrast to state-of-the-art implicit Eulerian formulations. The proposed formulation is approximate and departs from physical models in two ways. It is parameterized with a non-physical constant, and the vorticity diffusion is not encoded in the formulation of the target velocity gradient, but considered in the reconstruction process of the final velocity field. Experiments illustrate the close relation of our approach to true viscosity, its computational efficiency and also the large range of viscosities that can be handled. Complex scenarios with one- and two-way coupled solids and multiple fluid phases are presented.

2 Related work

Lagrangian techniques: Viscosity is an important stability aspect in SPH fluid simulations. The typically required discretiza-

*e-mail:peera@cs.uni-freiburg.de

ACM Reference Format
Peer, A., Ihmsen, M., Cornelis, J., Teschner, M. 2015. An Implicit Viscosity Formulation for SPH Fluids. ACM Trans. Graph. 34, 4, Article 114 (August 2015), 10 pages. DOI = 10.1145/2766925 http://doi.acm.org/10.1145/2766925.

SIGGRAPH '15 Technical Paper, August 09 – 13, 2015, Los Angeles, CA.

DOI: http://doi.acm.org/10.1145/2766925

tion of the Laplacian, however, is numerically challenging in SPH. [Müller et al. 2003] addressed this issue by introducing a specially designed kernel function. Alternatively, approximations of the second derivative by combining the first derivative with a finite difference approximation have been proposed [Morris et al. 1997], with Monaghan's *artificial viscosity* [Monaghan 1989] being a popular choice. Further, XSPH [Monaghan 1992; Monaghan 1989] implements a popular approximation of the Laplacian of the velocity field which is, e.g., employed in [Schechter and Bridson 2012] and [Macklin and Müller 2013]. For non-Newtonian fluids, viscosity forces can also be derived from the deformation tensor, e.g., [Paiva et al. 2006; Paiva et al. 2009; de Souza Andrade et al. 2014].

Although the usage of the Laplacian is the physically correct concept for incompressible fluids, there exist alternative approximate approaches to smooth the velocity field. E.g., [Stora et al. 1999] and [Steele et al. 2004] compute a viscosity force from relative velocities. Such approximate techniques significantly improve the stability of low viscous SPH fluids, but are less efficient for highly viscous fluids.

In the context of highly viscous fluids, the discussed explicit viscosity formulations suffer from small timesteps and are also prone to overshooting. Our approach differs in several aspects to address these issues. First, we use an implicit formulation which allows for significantly larger timesteps. Overshooting cannot occur. Second, we correctly account for desired velocity divergences that are computed by SPH pressure solvers to counteract density errors, whereas existing formulations typically assume a divergence-free velocity field. This is especially relevant for high viscosities and large timesteps.

The only implicit formulation for viscosity in SPH so far is [Takahashi et al. 2015], which adapts the Eulerian formulation of [Batty and Bridson 2008] for SPH. In contrast, our novel formulation bases on a target velocity gradient and does not interfere with the pressure solver. Thus, viscosity solver and pressure solver do not negatively affect each other. Our viscosity formulation does not require a final pressure solve which possibly perturbs the result of the viscosity solver.

Eulerian techniques: First approaches simply discretized the Laplacian [Foster and Metaxas 1996] and used explicit formulations, while a first implicit viscosity update has been introduced by [Stam 1999]. [Carlson et al. 2002] extended the implicit viscosity step to account for variable viscosity and for complex-shaped free surfaces. [Carlson et al. 2002] could also simulate melting materials, where the solid boundary handling was improved in [Fält and Roble 2003]. However, [Rasmussen et al. 2004] noted that the Laplacian-based formulation of [Carlson et al. 2002] is inconsistent for variable viscosities. Instead, they derived a consistent formulation using the strain-rate tensor. Due to the obtained non-symmetric system, they proposed an explicit-implicit scheme which could simulate impressive melting sequences.

Later, [Losasso et al. 2006] and [Batty and Bridson 2008] made the important observation that both, the Laplacian and the strain-rate formulations, only work on divergence-free velocity fields. Therefore, they perform pressure projections before and after the viscosity computation. [Batty and Bridson 2008] also improved the handling of free surfaces by avoiding an erroneous damping of rotational components. In contrast to previous works, their variational approach preserves rotational motions. They showed realistic coiling and buckling effects. Lastly, spatially adaptive [Batty and Houston 2011] and dimension-reduced [Batty et al. 2012] approaches have been proposed. For the later, also promising Lagrangian approaches, e.g., [Bergou et al. 2010], exist.

In contrast to all discussed Eulerian methods, our Lagrangian approach does not require any special treatment of the free surface to allow for rotational fluid movement. Further, our viscosity formulation does not only work on divergence-free velocity fields. As our formulation preserves arbitrary divergences that might have been introduced by the SPH pressure projection to account for density errors, a single pressure solve suffices. Differing from all discussed Eulerian formulations, we compute a desired velocity gradient that encodes viscous effects, does not affect rotation rates, and preserves divergence. From this velocity gradient, the respective velocity field is reconstructed.

Viscosity in multiphase fluids: In multiphase simulations, discontinuities at fluid interfaces have to handled. In Lagrangian approaches, this is typically done by averaging the viscosity constants, e.g., [Müller et al. 2005b; Solenthaler et al. 2007; Ren et al. 2014]. For Eulerian methods, [Hong and Kim 2005] proposed to extrapolate ghost velocities across the interface to counteract distortions in the velocity gradient caused by variable viscosities. The proposed implicit scheme could handle high viscosities and large viscosity differences. Later, [Losasso et al. 2006] showed how to handle viscosity at interfaces of fluids with different densities. They demonstrated impressive interactions of fluids with varying density and viscosity. Using the particle level set method, [Losasso et al. 2006] introduced additional costs for multiphase simulations, since on the interface between phases, a distinct level set for each phase has to be advected.

In contrast, our viscosity formulation is incorporated in [Solenthaler and Pajarola 2008] to handle multiple phases. It does not impose additional costs for multiple phases with different densities and viscosities. We solve the viscosity system for each phase independently and rely solely on pressure and cohesion forces for the interaction between phases.

Fluid-solid blending and viscoelastic fluids: Various approaches base on the observation that an infinitely viscous fluid is similar to a solid. These approaches generally blend fluid and solid dynamics according to the desired amount of viscosity, which can be achieved in various ways. E.g., the elastic solid update equation can be integrated into the fluid update [Keiser et al. 2005], shape matching can be used to derive rigid-body motions for adjacent particles [Takamatsu and Kanai 2011], forces can be computed which counteract the deviation of the predicted deformation from a given target deformation [Dagenais et al. 2012], or particle positions can be iteratively corrected in order to restore a previous particle configuration [Takahashi et al. 2014]. Recently, a hybrid Eulerian-Lagrangian Material Point Method has been proposed for simulating snow [Stomakhin et al. 2013]. The technique was later incorporated into a FLIP solver [Zhu and Bridson 2005] to achieve melting effects [Stomakhin et al. 2014].

These approaches are closely related to viscoelastic fluids, where additional elastic forces are introduced [Miller and Pearce 1989; Terzopoulos et al. 1991]. Alternatively, the Navier-Stokes equations are augmented with a term accounting for elastic forces, e.g. [Goktekin et al. 2004] for grids, [Wojtan and Turk 2008] for FEM or [Rafiee et al. 2007; Chang et al. 2009; Chang et al. 2011] for SPH. [Clavet et al. 2005] simulated viscoelastic material with spring-based viscosity forces that, however, do not consider shear rates.

In contrast to all these approaches, our method solely relies on the plain Navier-Stokes equation for fluids. Our approach does not account for rigid motion and elasticity. We also note that the topic of viscoelastic fluids opens towards the large topic of particle-based deformables, e.g., [Desbrun and Gascuel 1996; Müller et al. 2004; Becker et al. 2009; Gerszewski et al. 2009; Macklin et al. 2014],

whose discussion is beyond the scope of this paper.

Position-based methods: Our approach can be classified in two different ways. On one hand, the method is implicit as we solve a linear system for unknown velocities at the next timestep. On the other hand, we do not employ a classical force-based formulation with a standard physical parameter to build the linear system. Instead, we formulate a set of constraints that should be fulfilled at the next timestep which is typical for recent position-based methods PBD, e.g. [Bender et al. 2014b]. Another similarity to PBD is the usage of a non-standard parameter. Although our approach qualitatively damps shear rates in a viscous way, we employ a non-standard parameter in the range between zero and one. If considered as related to PBD, our approach could also be seen as a contribution to the development of PBD from geometric towards physical motivations. While early PBD, e.g. [Müller et al. 2005a; Rivers and James 2007], exclusively focused on geometric motivations, recent PBD research tends to focus on physical motivations. E.g., position-based fluids PBF [Macklin and Müller 2013] preserve incompressibility with a constraint that is linearly proportional to classical pressure and the computed displacements are clearly related to pressure forces. In the context of position-based elastic material, [Bender et al. 2014a] employs classical continuum-based formulations to compute strain energies. The paper further discusses the close relation of computed displacements to elastic forces computed with classical FEM. Similar in sense, our formulation imposes velocity constraints to damp shear rates which is the physical effect of shear viscosity.

3 Method

Viscosity forces are commonly computed from the divergence of a viscous stress tensor. The stress tensor τ is computed from the strain-rate tensor $\mathbf{D}$ which is the symmetric part of the velocity gradient $\nabla\mathbf{v}$, i.e., $\mathbf{D} = \frac{1}{2}(\nabla\mathbf{v} + (\nabla\mathbf{v})^T)$. Viscosity forces aim at minimizing all entries of the strain-rate tensor, where the actual effect depends on various aspects. It depends on the viscosity model, on the parameters that are employed in the computation of the stress tensor from the strain-rate tensor and on the timestep. This makes it rather involved to use viscosity forces for highly viscous fluids.

Instead of using explicit viscosity forces, we propose a novel implicit formulation. Based on a desired velocity gradient at the next timestep, we compute momentum-preserving velocities that correspond to the predicted gradient. Our scheme does not only predict a velocity gradient that obtains highly viscous material, but it also preserves the corrections of density deviations induced by the pressure solver. In contrast to existing schemes, pressure and viscosity solvers do not interfere. Velocity changes due to pressure forces are computed first. Then, the viscosity solver computes velocities that account for viscosity, but do not influence the rate of change of the volume. A final pressure solve, which possibly perturbs the result of the viscosity solver, is not required.

The following description starts with the employed decomposition of the velocity gradient. Then, we show how to compute the desired velocity gradient. Finally, we show how to reconstruct the particle velocities from the desired velocity gradients, i.e., how to compute momentum-preserving velocities to obtain a velocity field with the specified gradient at the next timestep.

3.1 Decomposition of the velocity gradient

We consider the decomposition of the velocity gradient into three components: the spin tensor $\mathbf{R}$, the expansion-rate tensor $\mathbf{V}$ and the shear-rate tensor $\mathbf{S}$:

$$\nabla\mathbf{v} = \underbrace{\frac{1}{2}\left(\nabla\mathbf{v} - (\nabla\mathbf{v})^T\right)}_{\mathbf{R}} + \underbrace{\frac{1}{3}(\nabla\cdot\mathbf{v})\mathbf{I}}_{\mathbf{V}} + \underbrace{\left(\frac{1}{2}\left(\nabla\mathbf{v} + (\nabla\mathbf{v})^T\right) - \frac{1}{3}(\nabla\cdot\mathbf{v})\mathbf{I}\right)}_{\mathbf{S}}. \quad (1)$$

The spin tensor $\mathbf{R}$ describes the rate of rotation at a particle, i.e., the vorticity. The expansion-rate tensor $\mathbf{V}$ describes the density change at a particle that has been computed by our pressure solver. This tensor can be used to realize bulk viscosity. It is typically not equal to zero in our IISPH implementation and preserved by the viscous solver. The shear-rate tensor $\mathbf{S}$ describes the rate of shear strain at a particle. This tensor is generally adapted to realize shear viscosity. Note that the expansion-rate and shear-rate tensors add to the strain-rate tensor $\mathbf{D} = \mathbf{V} + \mathbf{S}$. Our approach, however, requires the decomposition into expansion rate and shear rate. The divergence of the velocity field in $\mathbf{V}$ and $\mathbf{S}$ is computed as the trace of the velocity gradient: $\nabla\cdot\mathbf{v} = \mathrm{tr}(\nabla\mathbf{v})$.

3.2 Prediction of the velocity gradient

Our approach estimates the velocity gradient $\nabla\mathbf{v}$ at time t and its decomposition for all particles. The resulting components are adapted and summed-up to obtain a predicted velocity gradient $\nabla^\tau\mathbf{v}$ at time $t + \Delta t$. We distinguish different cases that basically depend on the particle density. Time and particle indices are omitted in this section to simplify the notation.

Particle density above rest density. Based on the decomposition of the current velocity gradient $\nabla\mathbf{v} = \mathbf{R} + \mathbf{V} + \mathbf{S}$, the predicted gradient at the next timestep is computed as

$$\nabla^\tau\mathbf{v} = \mathbf{R} + \mathbf{V} + \xi\mathbf{S} \quad (2)$$

with $0 \leq \xi \leq 1$ being our parameter for shear viscosity. If $\xi = 1$, the shear rate does not change, corresponding to minimal shear viscosity. For $\xi = 0$, the shear-rate tensor vanishes in the predicted velocity gradient, corresponding to a fluid with maximum shear viscosity. Intermediate values for ξ allow for different strengths of the desired shear viscosity. Note again that this formulation only affects the shear rate, while the rate of rotation and volume changes are not affected.

Preserving the expansion-rate tensor $\mathbf{V}$ is one of the key aspects in our formulation. SPH pressure solvers usually compute a velocity field that is not perfectly divergence-free [Becker and Teschner 2007; Solenthaler and Pajarola 2009; Macklin and Müller 2013; Ihmsen et al. 2014a]. Instead, some divergence is introduced to eliminate small density deviations. Therefore, the tensor $\mathbf{V}$ is not necessarily equal to zero. As the divergence computed by the pressure solver aims at minimizing the density error, we would like to preserve this divergence, i.e., the expansion-rate tensor $\mathbf{V}$. On one hand, this corresponds to zero bulk viscosity. On the other hand, however, it does not interfere with the result of the pressure solver. Therefore, our approach computes velocities that do not negatively affect the density error.

Physical viscosity diffuses vorticity. As our formulation preserves $\mathbf{R}$, this diffusion is not encoded in the predicted velocity gradient. The diffusion is, however, incorporated in the reconstruction process of the velocity field (see Section 3.3 and Eq. (4)).

Particle density below rest density. Here we consider two sub-cases. If the divergence of the velocity field is negative, the predicted gradient is computed as in the previous case: $\nabla^\tau\mathbf{v} = \mathbf{R} + \mathbf{V} + \xi\mathbf{S}$. If,

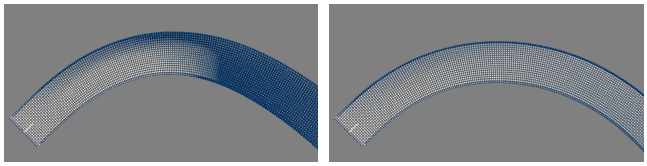


Figure 2: *Illustration of the divergence correction. In the left image, $\mathbf{V}$ is not altered. The density, color coded from white (rest density) to blue (below 70% rest density), drops. In the right image, $\mathbf{V}$ is removed from the target gradient if the particle density is below rest density to avoid the erroneous volume gain.*

however, the divergence is positive, we introduce maximum bulk viscosity by eliminating $\mathbf{V}$ from the predicted gradient:

$$\nabla^\tau \mathbf{v} = \mathbf{R} + \xi \mathbf{S}. \tag{3}$$

This formulation addresses an issue with negative pressure that is usually not considered in SPH pressure solvers. If the particle density is below rest density, this formulation aims at stopping adjacent particles from moving farther away by introducing maximum bulk viscosity. This reduces an artificial volume gain that otherwise could occur in our approach. The effect of this formulation is illustrated in Fig. 2. Note that this formulation is similar to forces from negative pressures. Adjacent particles, however, are not attracted, but only prevented from moving farther away.

3.3 Reconstruction of the velocity field

The final velocities $\mathbf{v}_i(t+\Delta t)$ are reconstructed from the target gradient $\nabla^\tau \mathbf{v}_i$. Therefore, the first-order Taylor approximation

$$\mathbf{v}_i(t+\Delta t) = \mathbf{v}_j(t+\Delta t) + \frac{\nabla^\tau \mathbf{v}_i + \nabla^\tau \mathbf{v}_j}{2}\mathbf{x}_{ij} \tag{4}$$

is considered for two particles i and j resulting in a linear system with unknown velocities. Here, $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ denotes the distance vector between the two particles. The average of the gradients $\nabla^\tau \mathbf{v}_i$ and $\nabla^\tau \mathbf{v}_j$ is used to guarantee momentum-preserving velocity changes at the particles. It also accounts for the diffusion of the rotation rate that is particularly present in highly viscous fluids. Employing Eq. (4) results in a novel implicit formulation for highly viscous fluids. Details of the linear system are described in Section 4.

3.4 Viscous SPH fluid solver

The proposed solver first computes an intermediate velocity $\mathbf{v}^*$ without considering pressure and viscosity. In the second stage, an updated intermediate velocity $\mathbf{v}^{**}$ is computed from pressure. We use IISPH [Ihmsen et al. 2014a]. However, alternative solvers, e.g., WCSPH [Becker and Teschner 2007], PCISPH [Solenthaler and Pajarola 2009] or PBF [Macklin and Müller 2013] could also be used. In the last stage, the proposed viscosity solver computes the final velocity $\mathbf{v}(t+\Delta t)$ and particles are advected with $\mathbf{v}(t+\Delta t)$. The viscosity solver does not affect the density correction induced by the pressure solver. Thus, in contrast to other techniques, a final pressure solve can be omitted. Algorithm 1 summarizes the proposed viscous SPH fluid solver.

Algorithm 1 Viscous SPH fluid solver

```
compute gravity and cohesion force
update velocity: v(t) → v*
compute pressure and pressure force
update velocity: v* → v**
solve viscosity system
update velocity: v** → v(t + Δt)
update position: x(t) → x(t + Δt)
```

4 Implementation

Velocity gradient: According to Algorithm 1, the viscosity solver starts with the velocity gradient $\nabla \mathbf{v}_i^{**}$ to compute the final velocities $\mathbf{v}_i(t+\Delta t)$. Following Monaghan's *second golden rule of SPH* [Monaghan 1992], we use the symmetric gradient expression

$$\nabla \mathbf{v}_i^{**} = \frac{1}{\rho_i}\sum_j m_j\left(\mathbf{v}_j^{**} - \mathbf{v}_i^{**}\right) \otimes \nabla W_{ij}, \tag{5}$$

with ρ and m being density and mass of the indexed particle, respectively. j is the set of particles within the support of particle i. $\nabla W_{ij} = \nabla W(\mathbf{x}_i - \mathbf{x}_j)$ is the kernel gradient that depends on the particle positions $\mathbf{x}$. Equation (5) reliably computes the gradient even at free surfaces and interfaces between different phases. It further correctly results in a zero gradient for constant fields. The velocity gradient $\nabla \mathbf{v}_i^{**}$ is modified to obtain the target gradient $\nabla^\tau \mathbf{v}_i^{**}$ as described in Section 3.2.

Linear system: Now, we aim at reconstructing the velocity field $\mathbf{v}_i(t+\Delta t)$ from this target gradient. Therefore, the relation in Eq. (4) is considered which would result in the following SPH formulation:

$$\mathbf{v}_i(t+\Delta t) = \sum_j V_j\left(\mathbf{v}_j(t+\Delta t) + \frac{\nabla^\tau \mathbf{v}_i^{**} + \nabla^\tau \mathbf{v}_j^{**}}{2}\mathbf{x}_{ij}\right)W_{ij}. \tag{6}$$

$V_j = \frac{m_j}{\rho_j}$ is the volume of particle j. The accuracy of this approximation, however, is significantly deteriorated at free surfaces due to the incomplete neighborhood. Therefore, we consider the normalized SPH interpolation

$$q_i = \frac{\sum_j V_j q_j W_{ij}}{\sum_j V_j W_{ij}} \tag{7}$$

for an arbitrary fluid quantity q_i and apply it to Eq. (6). To avoid the rather expensive computation of $\sum_j V_j W_{ij}$, we propose an approximate, more efficient normalization. All particles have the same mass m_0 and we assume that IISPH preserves a constant volume V_0 for all particles. Now, Eq. (7) can be transformed into

$$q_i \approx \frac{\sum_j V_0 q_j W_{ij}}{\sum_j V_0 W_{ij}} = \frac{\sum_j m_0 q_j W_{ij}}{\sum_j m_0 W_{ij}} \approx \frac{1}{\rho_i}\sum_j m_j q_j W_{ij} \tag{8}$$

which avoids the computation of $\sum_j V_j W_{ij}$. Now, the normalized form of Eq. (6) can efficiently be computed with

$$\mathbf{v}_i(t+\Delta t) = \frac{1}{\rho_i}\sum_j m_j\left(\mathbf{v}_j(t+\Delta t) + \frac{\nabla^\tau \mathbf{v}_i^{**} + \nabla^\tau \mathbf{v}_j^{**}}{2}\mathbf{x}_{ij}\right)W_{ij}. \tag{9}$$

This equation is considered at all particles, resulting in a linear system of unknown velocities $\mathbf{v}_i(t+\Delta t)$. In order to identify the coefficients a_{ij} of the the respective system matrix $\mathbf{A}$, we use the relation $\sum_j V_j q_j W_{ij} = V_i q_i W_{ii} + \sum_{j \neq i} V_j q_j W_{ij}$ and rewrite

Eq. (9) as

$$\Big(1-\frac{m_i}{\rho_i}W_{ii}\Big)\mathbf{v}_i(t+\Delta t)-\frac{1}{\rho_i}\sum_{j\neq i}m_j\mathbf{v}_j(t+\Delta t)W_{ij}=\frac{1}{\rho_i}\sum_j m_j\frac{\nabla^\tau\mathbf{v}_i^{**}+\nabla^\tau\mathbf{v}_j^{**}}{2}\mathbf{x}_{ij}W_{ij}. \quad (10)$$

This formulation is symmetrized by multiplication with ρ_i, yielding

$$\big(\rho_i-m_iW_{ii}\big)\mathbf{v}_i(t+\Delta t)-\sum_{j\neq i}m_j\mathbf{v}_j(t+\Delta t)W_{ij}=\sum_j m_j\frac{\nabla^\tau\mathbf{v}_i^{**}+\nabla^\tau\mathbf{v}_j^{**}}{2}\mathbf{x}_{ij}W_{ij}. \quad (11)$$

Now, we have three linear systems $\mathbf{A}\mathbf{v_x}=\mathbf{b_x}$, $\mathbf{A}\mathbf{v_y}=\mathbf{b_y}$, $\mathbf{A}\mathbf{v_z}=\mathbf{b_z}$ with $\mathbf{v_x},\mathbf{v_y},\mathbf{v_z}$ being the vector of all x-, y-, z-components of $\mathbf{v}(t+\Delta t)$ and $\mathbf{b_x},\mathbf{b_y},\mathbf{b_z}$ being the vector of all x-, y-, z-components of the right-hand side of Eq. (11), respectively. The components of $\mathbf{A}$ are $a_{ii}=\rho_i-m_iW_{ii}$ and $a_{ij}=-m_jW_{ij}$ according to the left-hand side of Eq. (11).

These three symmetric systems can be solved with various solvers. We use Conjugate Gradient and initialize the solver with $\mathbf{v}^{**}$. This prevents the generation of constant offsets, thus eliminating the problem that velocities are, e.g., damped out during free fall as mentioned in [Carlson et al. 2002]. We stop solving a system when the maximum residual is below a user-defined threshold typically in the order of $10^{-5}\rho_0$, with ρ_0 being the rest density of the fluid. The division by rest density reverts the change in magnitude introduced by multiplying the system with the density in Eq. (11) and ensures that the threshold is independent from the fluid density.

Boundary handling: The boundary handling is generally accomplished during the IISPH pressure solve with [Akinci et al. 2012b] as described in [Ihmsen et al. 2014a]. We further propose two types of additional boundary handling during the viscosity solve. Sticky boundaries damp shear rates between fluid and solid particles, while separating boundaries do not.

Sticky boundaries: Motivated by materials, such as honey, that tend to stick to solid objects, the sum of neighboring particles j in Eq. (9) is split into a sum of fluid neighbors $j\notin b$ and a sum of boundary neighbors b.

$$\mathbf{v}_i(t+\Delta t)=\frac{1}{\rho_i}\sum_{j\notin b}m_j\Big(\mathbf{v}_j(t+\Delta t)+\frac{\nabla^\tau\mathbf{v}_i^{**}+\nabla^\tau\mathbf{v}_j^{**}}{2}\mathbf{x}_{ij}\Big)W_{ij}+\frac{1}{\rho_i}\sum_b m_b\mathbf{v}_b^{**}W_{ib}. \quad (12)$$

This change only affects the right-hand side of the linear system in Eq. (11) and does not influence the efficiency of the solver.

Separating boundaries: With separating boundary conditions, the viscous fluid can flow freely, but not into the boundary. We therefore consider the divergence of a fluid particle i with respect to boundary particles b in each solver iteration l:

$$\nabla\cdot\mathbf{v}_i^l=\sum_b m_b\big(\mathbf{v}_i^l-\mathbf{v}_b^{**}\big)\nabla W_{ib} \quad (13)$$

with $\mathbf{v}_i^l$ being the velocity of particle i after l solver iterations. This divergence has to be larger or equal to zero in order to avoid fluid velocities into the boundary. If the divergence is negative, we search for a velocity $\hat{\mathbf{v}}_i^l$ with zero divergence with respect to the boundary, $\sum_b m_b\big(\hat{\mathbf{v}}_i^l-\mathbf{v}_b^{**}\big)\nabla W_{ib}=0$. The unknown velocity can be computed as

$$\hat{\mathbf{v}}_i^l=\frac{\mathbf{n}_i}{\mathbf{n}_i\cdot\mathbf{n}_i}\sum_b m_b\mathbf{v}_b^{**}\nabla W_{ib} \quad (14)$$

with $\mathbf{n}_i=\sum_b m_b\nabla W_{ib}$ which can be interpreted as the surface normal of the considered boundary particles. Adjusting the velocity in each solver iteration avoids fluid flow into the boundary. It is unfortunately not sufficient to adjust the final velocities. Thus, plain Conjugate Gradient does not work with separating boundaries and we fall back to Jacobi in this case.

Multiple phases: We incorporate the approach of [Solenthaler and Pajarola 2008] into IISPH to simulate multiple phases. Then, the viscosity system is solved independently for each phase. The interaction between the phases is accomplished via pressure and cohesion forces as in [Akinci et al. 2012b].

5 Discussion

Interpretation of the system: The left-hand side of Eq. (10) approximates the negative Laplacian of the unknown velocities $\mathbf{v}_i(t+\Delta t)$. Interestingly, the right-hand side approximates the same negative Laplacian, but with known values incorporating the target gradient. This can be seen after substituting $\frac{\nabla^\tau\mathbf{v}_i^{**}+\nabla^\tau\mathbf{v}_j^{**}}{2}\mathbf{x}_{ij}$ with $\mathbf{v}_i(t+\Delta t)-\mathbf{v}_j(t+\Delta t)$ according to Eq. (4). Thus, the right-hand side constitutes a target Laplacian built from the target gradient. I.e., Eq. (10) actually estimates velocities that meet an approximated target Laplacian. So, the formulation is conceptually different to the standard implicit formulation $(\mathbf{I}-\Delta t\mu\nabla^2)\mathbf{v}_i(t+\Delta t)=\mathbf{v}_i^{**}$. Instead, Eq. (10) can be interpreted as $\nabla^2\mathbf{v}_i(t+\Delta t)=\nabla^{2^\tau}\mathbf{v}_i^{**}$ with $\nabla^{2^\tau}\mathbf{v}_i^{**}$ being the desired Laplacian that is approximated from the target velocity gradient $\nabla^\tau\mathbf{v}_i^{**}$ which is computed from known velocities $\mathbf{v}_i^{**}$.

Parameter: The parameter ξ is a non-standard parameter and does not correspond to dynamic or kinematic viscosity. Nevertheless, ξ is physically motivated. It governs the damping of the shear rate in the fluid, thus it governs viscosity.

It is also important to note that ξ depends on the timestep and the resolution which is illustrated in Section 6, but not further addressed in this paper. Our approach shares this issue with PBD, where the time dependency of parameters is not handled as timesteps typically do not significantly vary in a scenario (e.g., [Bender et al. 2014b]). One of the first papers that discuss resolution-dependent parameters in SPH is, e.g., [Akinci et al. 2013]. However, timestep dependencies are not addressed.

Limitations: The proposed concept does not explicitly alter spin. Furthermore, it preserves divergence, i.e., incompressibility and shear viscosity are decoupled which is novel. The discretized implementation, however, employs various numerical approximations. E.g., an SPH approximation is used for the velocity gradient $\nabla^\tau\mathbf{v}_i^{**}$, and Eq. (4) employs a Taylor approximation with an averaged velocity gradient. This results in numerical errors in spin, divergence, target gradient, and momentum. These errors, however, are rather small as indicated by the large timesteps and the high viscosities that can be obtained with our approach. In this context, Section 6 discusses an experiment that shows that errors in the preservation of incompressibility are negligible. The error in spin also accounts for the physical effect of vorticity diffusion in highly viscous fluids.

Zero viscosity, i.e. non-destructive velocity reconstruction works in our approach, as the solver is initialized with $\mathbf{v}_i^{**}$ and performs zero

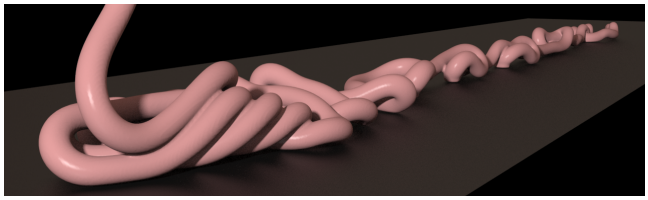


Figure 3: *Realistic buckling and coiling with up to 100k particles and an average computation time of 5s per frame.*

iterations, if the actual and the target velocity gradient are equal. However, due to the smoothing of the velocity field in the reconstruction process, our approach should not be used for low viscous fluids such as water.

Our approach is rather expensive for rigid-like materials where it requires many solver iterations. It counteracts shear rates, but has no notion of shear. Thus, it suffers from drifting shear that cannot be corrected. So, conceptually, viscoelastic models should be preferred for rigid-like objects.

Physical viscosity diffuses vorticity. This aspect is omitted in the formulation of the predicted velocity gradient. Although the averaging of the velocity gradients in Eq. (4) accounts for such a diffusion, the relation of the introduced diffusion to the diffusion in other viscosity approaches or in true viscosity remains unclear.

6 Results

This section illustrates the properties of the proposed viscosity solver. For the presented experiments, the proposed viscosity solver is combined with an IISPH pressure solver [Ihmsen et al. 2014a]. We employ the cubic spline kernel [Monaghan 2005] with a smoothing length of twice the particle distance. The maximum overall volume deviation in all scenarios is below 0.1%. This volume deviation is guaranteed by the IISPH pressure solver and generally not negatively influenced by the proposed viscosity solver. [Becker and Teschner 2007] is used for surface tension. One-way and two-way coupled boundary handling is realized with [Akinci et al. 2012b; Ihmsen et al. 2010]. Our implementation is fully parallelized [Ihmsen et al. 2011]. Surfaces are reconstructed with [Mootz 2014] and [Ju et al. 2002]. Alternatively, [Akinci et al. 2012a] could be used. The images and video sequences were rendered with Houdini's Mantra renderer [Side Effects Software 2013] with 50 frames per second. All experiments have been computed on a 24-core 2.70 GHz Intel Xeon workstation.

Buckling and coiling: The setting in Fig. 3 illustrates that the proposed viscosity solver captures the spontaneous buckling and coiling effects that characterize highly viscous fluids. The timestep is 0.002s for a particle spacing of 0.04m. Neighborhood and pressure require 25ms per step, viscosity requires 195ms with an average of 50 solver iterations.

Viscosity range: Fig. 4 shows two frames at the same time of simulations with different viscosity parameters. The timestep is 0.001s for a particle spacing of 0.012m. The computation time of the viscous solver scales with the viscosity parameter ξ. The accompanying video illustrates that the approach can handle highly viscous material. The experiment in Fig. 4 also illustrates the fact that the result and thus, the efficiency of the pressure solver are not negatively affected by the viscosity solver. While the viscosity solver requires significantly differing iterations for different viscosity parameters, the efficiency of the pressure solve does not depend on the viscosity. The viscosity solver requires 25 iterations for $\xi = 0.8$ and 87 iterations for $\xi = 0.2$, while the pressure solver requires 5 iterations in both settings.

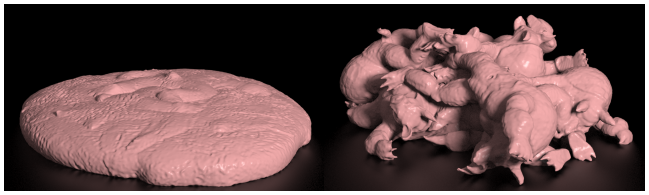


Figure 4: *Different viscosities. Both scenarios consist of 1.3M particles. In the low viscous scenario with $\xi = 0.8$, the average computation time per frame is 11s, while the highly viscous setting with $\xi = 0.2$ required an average computation time per frame of 37s.*

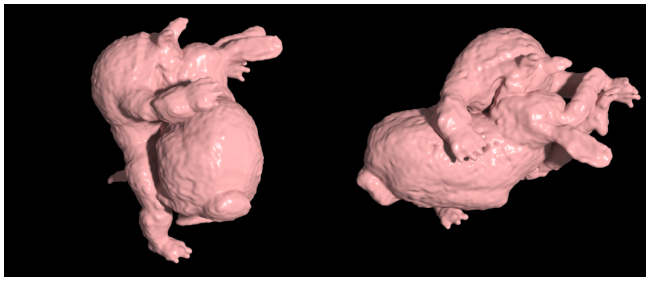


Figure 5: *Rotation rates are properly handled. The scene consists of 140k particles. The average computation time per frame is 1.7s*

Rotation rate and density error: The proposed viscosity solver properly handles rotation rates and does not affect the density error. This is illustrated in Fig. 5 with two colliding objects in a zero-gravity environment. The timestep is 0.001s for a particle spacing of 0.02m. The viscosity parameter is $\xi = 0.5$, resulting in 17 iterations per viscous solve. The IISPH pressure solve requires 3 iterations. Fig. 6 illustrates the densities before and after the viscous solve for the same scenario. The graph indicates that the density deviation introduced by the viscosity solver is negligible.

Time-varying viscosity: Fig. 7 illustrates a scenario with time-varying viscosity. The scene is simulated with $\xi = 0$ for some time for all Armadillos. Then, for the blue Armadillo, ξ is raised from 0 to 0.99 instantaneously. For the yellow and the red Armadillo, ξ is linearly raised from 0 to 0.99 within 3s and 15s, respectively. The scenario illustrates that rigid-like objects and fluids with moderately low viscosity can be simulated with our approach. However, as discussed in Section 5, our approach is less appropriate for low

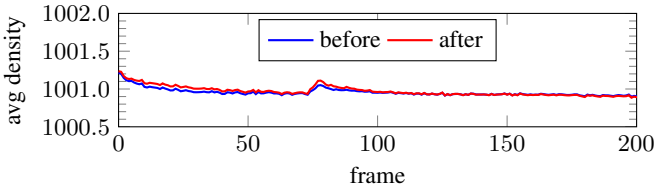


Figure 6: *Densities before and after the viscosity solve for the first two hundred frames of the scene in Fig. 5. It can be seen that the effect of our viscous solver on the density error is negligible.*

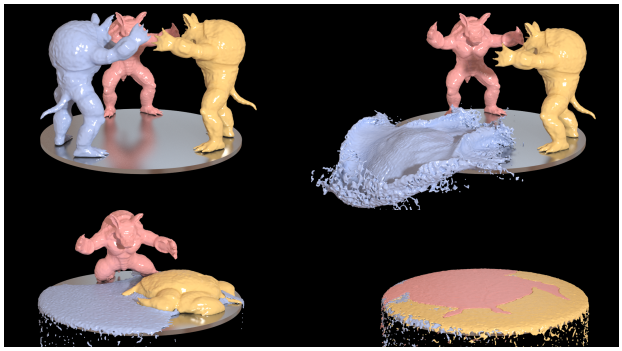


Figure 7: *Objects with time-varying viscosity:* $0 \leq \xi \leq 0.99$. *The average computation time per frame is 18s for 390k particles.*

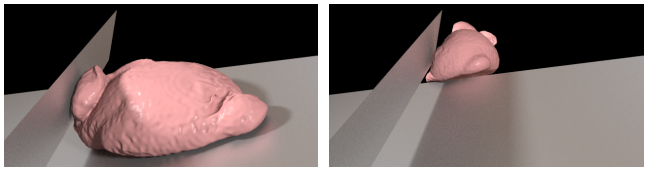


Figure 8: *Sticky (left) and separating (right) boundary handling. In both scenarios, 100k particles are used.*

viscous fluids such as water. For rigid-like material, our viscosity solver is certainly less efficient than viscoelastic solvers.

Boundary handling: Fig. 8 illustrates sticky and separating boundary conditions. The timestep is 0.002s for a particle spacing of 0.025m. The viscosity is $\xi = 0.5$. For sticky boundaries, the viscous solver requires 17 iterations. As Conjugate Gradient is not used for separating boundaries, this setting requires 230 Jacobi iterations. Sticky boundaries would also require 230 Jacobi iterations, illustrating the obvious fact that Conjugate Gradient with 17 iterations is more efficient for sticky boundaries.

Complex boundaries: The scenario in Fig. 1 illustrates the one-way coupled interaction with complex moving boundaries. The timestep is 0.0004s for a particle spacing of 0.02m. With $\xi = 0.5$, the viscous solver requires 12 iterations.

Two-way coupling: Fig. 9 shows a scene where a viscous fluid and a fracturing solid are two-way coupled. The timestep is 0.0005s for a particle spacing of 0.014m. With a viscosity of $\xi = 0.8$, the viscous solver requires 9 iterations.

Multiple phases: The proposed viscosity solver is particularly appropriate for multiple phases as shown in Fig. 10 with five interacting phases with a viscosity range from $\xi = 1$ to $\xi = 0$. Water can be simulated with $\xi = 1$ or by simply omitting the viscous solve. The interfaces between different phases are accurately handled, as surface particles of a phase only consider velocities of particles from the same phase in the predicted velocity gradient. The densities in Fig. 10 range from $100 \frac{kg}{m^3}$ to $2000 \frac{kg}{m^3}$. Depending on the viscosity parameter, the viscous solve requires between 12 and 140 iterations. For a particle distance of 0.05m, a timestep of 0.001s is required.

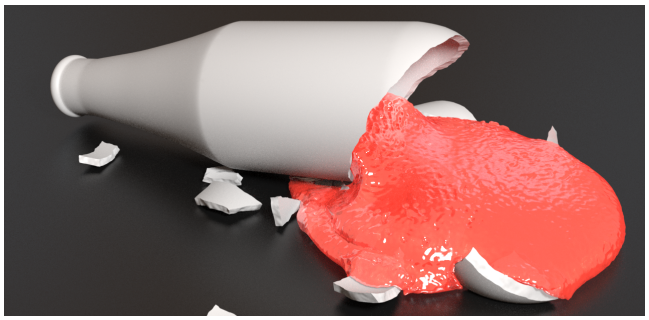


Figure 9: *Two-way coupling. The fluid consists of 570k particles. The computation time is 21s per frame.*

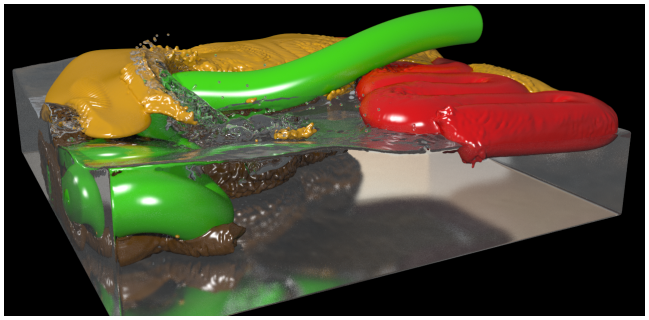


Figure 10: *Multiple phases with up to 3.1M particles. The average computation time per frame is 79s.*

Large-scale scenarios: Fig. 11 shows a scene with 11M particles. The particle distance is 0.25m with a timestep of 0.04s. Pressure and viscosity require up to 10 and 30 iterations, respectively.

Explicit and implicit viscosity solvers: A thorough performance comparison of the proposed viscosity solver with existing approaches is beyond the scope of this paper and also not possible within the given constraints. E.g., our approach guarantees a maximum volume deviation, while others have no notion of the actual volume deviation. Our approach requires one pressure solve, others require two. It is also involved to parameterize different approaches such that the resulting dynamic viscous behaviors match. To at least indicate that our approach is superior to explicit formulations, we show a comparison with artificial viscosity [Monaghan 1989] in Fig. 12.

A second scenario for comparing our approach with an explicit viscosity formulation is illustrated in Fig. 13. The fluid is initialized in a static cylinder. Then, the cylinder starts rotating with constant velocity. After the fluid velocity field has reached an equilibrium state, the cylinder stops rotating. The experiment has been performed with artificial viscosity [Monaghan 1989] for $\mu = 50$ (oil) and with the proposed viscosity approach for $\xi = 0$ and $\xi = 0.9$. Artificial viscosity and our approach with $\xi = 0.9$ have been simulated in two resolutions.

For all settings, the same velocity field (Fig. 13 (c)) is achieved for the rotating cylinder and the same velocity field (Fig. 13 (b)) is achieved after stopping the cylinder. This means that our approach conceptually corresponds to shear viscosity, i.e., velocity changes are induced as long as shear rates are present in the fluid.

The actual viscosity, i.e., the speed at which the equilibrium is reached, differs in all experiments and we did not investigate, how

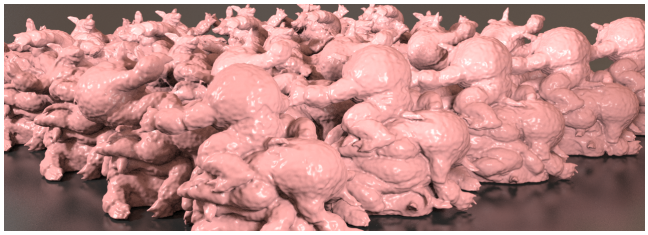


Figure 11: *Large-scale scenario with 11M particles and $\xi = 0.5$. The average computation time per frame is 144s.*

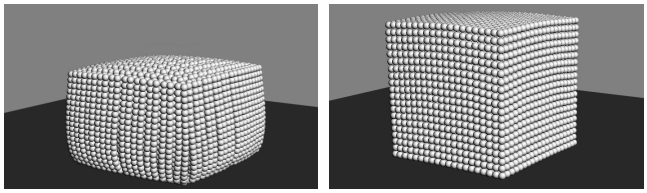


Figure 12: *Comparison of artificial viscosity [Monaghan 1989] on the left-hand side with our approach on the right-hand side. 8k particles with a spacing of 0.05m are used in both experiments. The overall computation is 20min for the explicit formulation, while our implicit formulation required 11s and also results in a more viscous fluid. This difference is mainly due to the obvious fact that the explicit formulation required a rather small timestep of 0.00002s, while our formulation could be simulated with a timestep of 0.01s.*

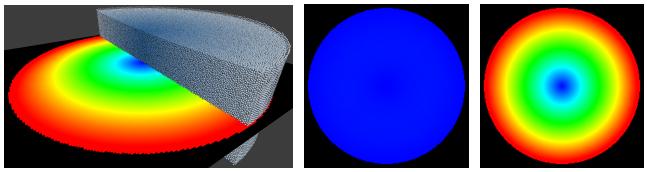
to map our non-physical constant ξ to dynamic viscosity μ. Obviously, implicit approaches have the capability to obtain the equilibrium faster than explicit approaches as the shear rates - introduced by accelerating and stopping the cylinder - are propagated within solver iterations instead of per simulation step. In particular, the equilibrium is immediately achieved for $\xi = 0$ within one timestep. This property is also a reason why low viscosities are difficult to realize with implicit solvers.

Fig. 14 shows that the viscosity varies in all five settings. This is obvious for $\mu = 50$, $\xi = 0$ and $\xi = 0.9$. The images also illustrate that our approach results in different viscosities for the same parameter, when timestep and resolution change. I.e., the constant ξ depends on timestep and resolution which is not further investigated in this paper. It is further interesting to note that the differences in Fig. 14 (a) and (b) indicate that the effect of Monaghan's explicit formulation also depends on timestep and resolution for a constant dynamic viscosity parameter.

Discussion: This paper proposes a novel efficient and versatile concept for simulating highly viscous fluids, but it does not introduce a novel material model. As the minimization of the shear rate is standard in most viscosity solvers, e.g. [Batty and Bridson 2008; Takahashi et al. 2015], it is beyond the scope of this paper to analyze the relation of the employed material model to real-world experiments.

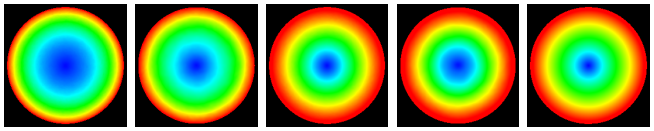
7 Conclusion and future work

We have presented a novel implicit formulation for highly viscous SPH fluids that is based on prescribing a target gradient and reconstructing the corresponding velocity field. By ensuring that the target gradient only changes the shear rate, but not the divergence, our viscosity formulation preserves the density correction induced by



(a) **(b)** **(c)**

Figure 13: *Fluid in an invisible cylinder with color-coded velocities (a). Blue and white particles have min and max velocities, respectively. Half of the fluid is not visualized to illustrate the employed sensor plane where blue and red indicate min and max velocities, respectively. The fluid is initialized in a static cylinder corresponding to the velocity field in (b). Then, the cylinder starts rotating which results in the velocity field in (c). Stopping the cylinder finally results in the original velocity field (b).*



(a) **(b)** **(c)** **(d)** **(e)**

Figure 14: *Intermediate velocity fields for different settings at the same time after starting the cylinder rotation: (a) artificial viscosity $\mu = 50$, (b) artificial viscosity $\mu = 50$ at half resolution and with doubled timestep, (c) our viscosity $\xi = 0.9$, (d) our viscosity $\xi = 0.9$ at half resolution and with doubled timestep, (e) our viscosity $\xi = 0$. These intermediate states differ due to different viscosities. Note that all experiments finally reach the same equilibrium state shown in Fig. 13 (c).*

the SPH pressure solver. This is achieved by decomposing the gradient into expansion-rate, shear-rate and spin tensors. Our formulation does not require a divergence-free velocity field as input. It can be added to common SPH solvers as a post-processing step. By integrating our formulation into a state-of-the-art SPH solver such as IISPH, a maximum volume deviation, typically below 0.1%, can be guaranteed independent from the grade of viscosity.

We showed that our formulation is significantly more efficient than explicit methods. It also achieves a larger range of viscosities. The versatility of our approach was demonstrated by simulating a wide range of viscous materials. We also showed that it can be easily integrated into one- and two-way coupling scenarios as well as into multiphase simulations, which illustrates the usability of our formulation.

Future work could address some potential improvements. Since our method heavily damps shear rates, particle movement is strongly restricted. Therefore, even after several simulation seconds, the pattern of the initial particle sampling is still visible. This is especially problematic for regular sampling patterns. Investigation into random sampling patterns, such as Poisson disk sampling, could be interesting.

In our implementation, we solve the linear system with Conjugate Gradient. This works well for sticky boundary condition, but cannot be used for separating boundary conditions, where intermediate results have to be adjusted in each solver iteration. Fortunately, most of the viscous materials tend to stick to solid boundaries. Still, an investigation into a more efficient solver for the proposed separating boundary conditions could be worthwhile.

Acknowledgements

This project is supported by the German Research Foundation (DFG) under contract number TE 632/1. We would like to thank Christoph Gissler and Andreas Henne for supporting the project and the reviewers for providing impulses that helped to improve the manuscript. The Armadillo and Bunny models are courtesy of the Stanford Computer Graphics Laboratory. The hand mixer model is courtesy of Daniel Isler. The bottle model is courtesy of genx473 at www.blendswap.com.

References

AKINCI, G., IHMSEN, M., AKINCI, N., AND TESCHNER, M. 2012. Parallel surface reconstruction for particle-based fluids. *Computer Graphics Forum 31*, 6, 1797–1809.

AKINCI, N., IHMSEN, M., AKINCI, G., SOLENTHALER, B., AND TESCHNER, M. 2012. Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics (TOG) 31*, 4, 62.

AKINCI, N., AKINCI, G., AND TESCHNER, M. 2013. Versatile surface tension and adhesion for SPH fluids. *ACM Trans. Graph. 32*, 6 (Nov.), 182:1–182:8.

BATTY, C., AND BRIDSON, R. 2008. Accurate viscous free surfaces for buckling, coiling, and rotating liquids. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics symposium on computer animation*, Eurographics Association, 219–228.

BATTY, C., AND HOUSTON, B. 2011. A simple finite volume method for adaptive viscous liquids. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, 111–118.

BATTY, C., URIBE, A., AUDOLY, B., AND GRINSPUN, E. 2012. Discrete viscous sheets. *ACM Transactions on Graphics (TOG) 31*, 4, 113.

BECKER, M., AND TESCHNER, M. 2007. Weakly compressible SPH for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, 209–217.

BECKER, M., IHMSEN, M., AND TESCHNER, M. 2009. Corotated SPH for deformable solids. In *Proceedings of the Fifth Eurographics conference on Natural Phenomena*, Eurographics Association, 27–34.

BENDER, J., KOSCHIER, D., CHARRIER, P., AND WEBER, D. 2014. Position-based simulation of continuous materials. *Computers & Graphics 44*, 0, 1 – 10.

BENDER, J., MÜLLER, M., OTADUY, M. A., TESCHNER, M., AND MACKLIN, M. 2014. A survey on position-based simulation methods in computer graphics. *Computer Graphics Forum 33*, 6, 228–251.

BERGOU, M., AUDOLY, B., VOUGA, E., WARDETZKY, M., AND GRINSPUN, E. 2010. Discrete viscous threads. *ACM Transactions on Graphics (TOG) 29*, 4, 116.

CARLSON, M., MUCHA, P. J., VAN HORN III, R. B., AND TURK, G. 2002. Melting and flowing. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM, 167–174.

CHANG, Y., BAO, K., LIU, Y., ZHU, J., AND WU, E. 2009. A particle-based method for viscoelastic fluids animation. In *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, ACM, 111–117.

CHANG, Y., BAO, K., ZHU, J., AND WU, E. 2011. High viscosity fluid simulation using particle-based method. In *VR Innovation (ISVRI), 2011 IEEE International Symposium on*, IEEE, 199–205.

CLAVET, S., BEAUDOIN, P., AND POULIN, P. 2005. Particle-based viscoelastic fluid simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM, 219–228.

DAGENAIS, F., GAGNON, J., AND PAQUETTE, E. 2012. A prediction-correction approach for stable SPH fluid simulation from liquid to rigid. *Proceedings of the Computer Graphics International 2012*.

DE SOUZA ANDRADE, L. F., SANDIM, M., PETRONETTO, F., PAGLIOSA, P. A., AND PAIVA, A. 2014. SPH fluids for viscous jet buckling. In *27th SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI 2014*, SBC, 65–72.

DESBRUN, M., AND GASCUEL, M.-P. 1996. Smoothed particles: A new paradigm for animating highly deformable bodies. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation '96*, Springer-Verlag, Eurographics, 61–76.

FÄLT, H., AND ROBLE, D. 2003. Fluids with extreme viscosity. In *ACM SIGGRAPH 2003 Sketches & Applications*, ACM, 1–1.

FOSTER, N., AND METAXAS, D. 1996. Realistic animation of liquids. *Graphical models and image processing 58*, 5, 471–483.

GERSZEWSKI, D., BHATTACHARYA, H., AND BARGTEIL, A. W. 2009. A point-based method for animating elastoplastic solids. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, 133–138.

GOKTEKIN, T. G., BARGTEIL, A. W., AND O'BRIEN, J. F. 2004. A method for animating viscoelastic fluids. *ACM Transactions on Graphics (TOG) 23*, 3, 463–468.

HONG, J.-M., AND KIM, C.-H. 2005. Discontinuous fluids. *ACM Transactions on Graphics (TOG) 24*, 3, 915–920.

IHMSEN, M., AKINCI, N., GISSLER, M., AND TESCHNER, M. 2010. Boundary handling and adaptive time-stepping for PCISPH. In *Proceedings VRIPHYS*, VRIPHYS, 79–88.

IHMSEN, M., AKINCI, N., BECKER, M., AND TESCHNER, M. 2011. A parallel SPH implementation on multi-core CPUs. *Computer Graphics Forum 30*, 1, 99–112.

IHMSEN, M., CORNELIS, J., SOLENTHALER, B., HORVATH, C., AND TESCHNER, M. 2014. Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics 20*, 3 (Mar.), 426–435.

IHMSEN, M., ORTHMANN, J., SOLENTHALER, B., KOLB, A., AND TESCHNER, M. 2014. SPH fluids in computer graphics. In *Eurographics 2014-State of the Art Reports*, The Eurographics Association, 21–42.

JU, T., LOSASSO, F., SCHAEFER, S., AND WARREN, J. 2002. Dual contouring of hermite data. *ACM Transactions on Graphics (TOG) 21*, 3, 339–346.

KEISER, R., ADAMS, B., GASSER, D., BAZZI, P., DUTRÉ, P., AND GROSS, M. 2005. A unified Lagrangian approach to solid-fluid animation. In *Point-Based Graphics, 2005. Eurographics/IEEE VGTC Symposium Proceedings*, IEEE, 125–148.

LOSASSO, F., SHINAR, T., SELLE, A., AND FEDKIW, R. 2006. Multiple interacting liquids. *ACM Transactions on Graphics (TOG) 25*, 3, 812–819.

MACKLIN, M., AND MÜLLER, M. 2013. Position based fluids. *ACM Transactions on Graphics (TOG) 32*, 4, 104.

MACKLIN, M., MÜLLER, M., CHENTANEZ, N., AND KIM, T.-Y. 2014. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG) 33*, 4, 104.

MILLER, G., AND PEARCE, A. 1989. Globular dynamics: A connected particle system for animating viscous fluids. *Computers & Graphics 13*, 3, 305–309.

MONAGHAN, J. 1989. On the problem of penetration in particle methods. *Journal of Computational physics 82*, 1, 1–15.

MONAGHAN, J. J. 1992. Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics 30*, 543–574.

MONAGHAN, J. J. 2005. Smoothed particle hydrodynamics. *Reports on progress in physics 68*, 8, 1703.

MOOTZ, E., 2014. emPolygonizer5. `http://www.mootzoid.com/`.

MORRIS, J. P., FOX, P. J., AND ZHU, Y. 1997. Modeling low reynolds number incompressible flows using SPH. *Journal of computational physics 136*, 1, 214–226.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, 154–159.

MÜLLER, M., KEISER, R., NEALEN, A., PAULY, M., GROSS, M., AND ALEXA, M. 2004. Point based animation of elastic, plastic and melting objects. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, 141–151.

MÜLLER, M., HEIDELBERGER, B., TESCHNER, M., AND GROSS, M. 2005. Meshless deformations based on shape matching. In *ACM SIGGRAPH 2005 Papers*, ACM, New York, NY, USA, SIGGRAPH '05, ACM, 471–478.

MÜLLER, M., SOLENTHALER, B., KEISER, R., AND GROSS, M. 2005. Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM, 237–244.

PAIVA, A., PETRONETTO, F., LEWINER, T., AND TAVARES, G. 2006. Particle-based non-Newtonian fluid animation for melting objects. In *Computer Graphics and Image Processing, 2006. SIBGRAPI'06. 19th Brazilian Symposium on*, IEEE, 78–85.

PAIVA, A., PETRONETTO, F., LEWINER, T., AND TAVARES, G. 2009. Particle-based viscoplastic fluid/solid simulation. *Computer-Aided Design 41*, 4, 306–314.

RAFIEE, A., MANZARI, M., AND HOSSEINI, M. 2007. An incompressible SPH method for simulation of unsteady viscoelastic free-surface flows. *International Journal of Non-Linear Mechanics 42*, 10, 1210–1223.

RASMUSSEN, N., ENRIGHT, D., NGUYEN, D., MARINO, S., SUMNER, N., GEIGER, W., HOON, S., AND FEDKIW, R. 2004. Directable photorealistic liquids. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, 193–202.

REN, B., LI, C., YAN, X., LIN, M. C., BONET, J., AND HU, S.-M. 2014. Multiple-fluid SPH simulation using a mixture model. *ACM Transactions on Graphics (TOG) 33*, 5, 171.

RIVERS, A. R., AND JAMES, D. L. 2007. Fastlsm: Fast lattice shape matching for robust real-time deformation. In *ACM SIGGRAPH 2007 Papers*, ACM, New York, NY, USA, SIGGRAPH '07, ACM.

SCHECHTER, H., AND BRIDSON, R. 2012. Ghost SPH for animating water. *ACM Transactions on Graphics (TOG) 31*, 4, 61.

SIDE EFFECTS SOFTWARE, 2013. Houdini. `http://www.sidefx.com/`.

SOLENTHALER, B., AND PAJAROLA, R. 2008. Density contrast SPH interfaces. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics symposium on computer animation*, Eurographics Association, 211–218.

SOLENTHALER, B., AND PAJAROLA, R. 2009. Predictive-corrective incompressible SPH. *ACM Transactions on Graphics (TOG) 28*, 3, 40.

SOLENTHALER, B., SCHLÄFLI, J., AND PAJAROLA, R. 2007. A unified particle model for fluid–solid interactions. *Computer Animation and Virtual Worlds 18*, 1, 69–82.

STAM, J. 1999. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 121–128.

STEELE, K., CLINE, D., EGBERT, P. K., AND DINERSTEIN, J. 2004. Modeling and rendering viscous liquids. *Computer Animation and Virtual Worlds 15*, 3-4, 183–192.

STOMAKHIN, A., SCHROEDER, C., CHAI, L., TERAN, J., AND SELLE, A. 2013. A material point method for snow simulation. *ACM Transactions on Graphics (TOG) 32*, 4, 102.

STOMAKHIN, A., SCHROEDER, C., JIANG, C., CHAI, L., TERAN, J., AND SELLE, A. 2014. Augmented MPM for phase-change and varied materials. *ACM Transactions on Graphics (TOG) 33*, 4, 138.

STORA, D., AGLIATI, P.-O., CANI, M.-P., NEYRET, F., GASCUEL, J.-D., ET AL. 1999. Animating lava flows. In *Graphics Interface (GI'99) Proceedings*, GI, 203–210.

TAKAHASHI, T., NISHITA, T., AND FUJISHIRO, I. 2014. Fast simulation of viscous fluids with elasticity and thermal conductivity using position-based dynamics. *Computers & Graphics 43*, 0, 21 – 30.

TAKAHASHI, T., DOBASHI, Y., FUJISHIRO, I., NISHITA, T., AND LIN, M. C. 2015. Implicit formulation for SPH-based viscous fluids. *Computer Graphics Forum 34*, 2.

TAKAMATSU, K., AND KANAI, T. 2011. A fast and practical method for animating particle-based viscoelastic fluids. *International Journal of Virtual Reality 10*, 1, 29.

TERZOPOULOS, D., PLATT, J., AND FLEISCHER, K. 1991. Heating and melting deformable models. *The Journal of Visualization and Computer Animation 2*, 2, 68–73.

WOJTAN, C., AND TURK, G. 2008. Fast viscoelastic behavior with thin features. *ACM Transactions on Graphics (TOG) 27*, 3, 47.

ZHU, Y., AND BRIDSON, R. 2005. Animating sand as a fluid. *ACM Transactions on Graphics (TOG) 24*, 3, 965–972.