EUROGRAPHICS 2018 / D. Gutierrez and A. Sheffer
(Guest Editors)

Volume 37 (2018), Number 2

A Physically Consistent Implicit Viscosity Solver for SPH Fluids

Marcel Weiler[1], Dan Koschier[2], Magnus Brand[2], and Jan Bender[2]

[1]Graduate School CE, Technische Universität Darmstadt
[2]RWTH Aachen University

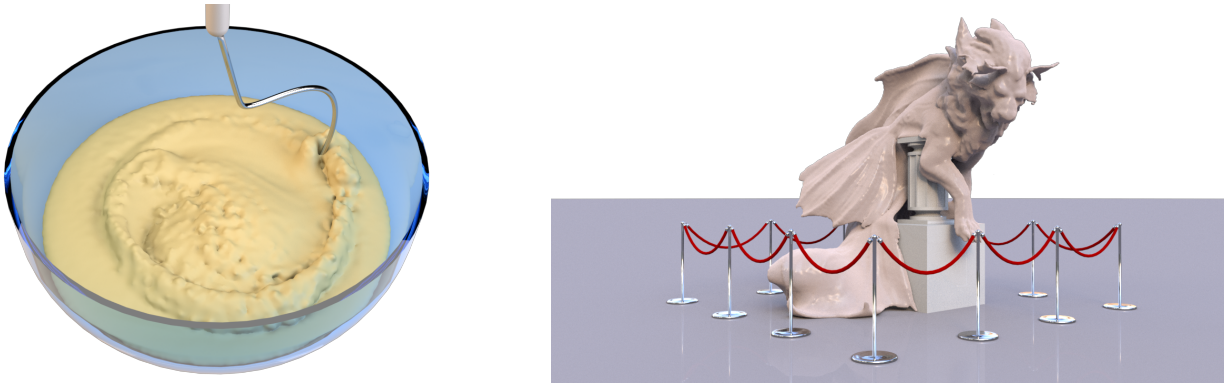


Figure 1: Our novel implicit viscosity solver allows the physically consistent simulation of highly viscous fluids. Left: 406k particles interacting with a fast rotating dough hook. Right: Large-scale simulation of a viscous statue consisting of 1.1M particles.

Abstract

In this paper, we present a novel physically consistent implicit solver for the simulation of highly viscous fluids using the Smoothed Particle Hydrodynamics (SPH) formalism. Our method is the result of a theoretical and practical in-depth analysis of the most recent implicit SPH solvers for viscous materials. Based on our findings, we developed a list of requirements that are vital to produce a realistic motion of a viscous fluid. These essential requirements include momentum conservation, a physically meaningful behavior under temporal and spatial refinement, the absence of ghost forces induced by spurious viscosities and the ability to reproduce complex physical effects that can be observed in nature. On the basis of several theoretical analyses, quantitative academic comparisons and complex visual experiments we show that none of the recent approaches is able to satisfy all requirements. In contrast, our proposed method meets all demands and therefore produces realistic animations in highly complex scenarios. We demonstrate that our solver outperforms former approaches in terms of physical accuracy and memory consumption while it is comparable in terms of computational performance. In addition to the implicit viscosity solver, we present a method to simulate melting objects. Therefore, we generalize the viscosity model to a spatially varying viscosity field and provide an SPH discretization of the heat equation.

CCS Concepts

•***Computing methodologies*** → ***Physical simulation;***

1. Introduction

Fluid motion is responsible for many interesting effects that capture the attention of the viewer. Inviscid fluids splash and swirl around, often change topology and thus provide rich detail at the fluid's surface. Viscous fluids on the other hand bridge the gap between fluids and solids. They try to maintain their shape after interacting with other objects, resist external forces and show the buckling and rope coiling effects that can be observed when a thin stream of honey or caramel flows from a spoon.

While the simulation of inviscid flow using Smoothed Particle Hydrodynamics (SPH) has been widely addressed, the simulation of incompressible, highly viscous fluids still poses major challenges. Methods based on explicit time integration of viscous forces suffer from strict time step restrictions to maintain stability. Mod-

ern approaches therefore suggest to use implicit viscosity solvers that allow the usage of large time steps while keeping the simulation stable. Very recently, several SPH-based implicit solvers to simulate highly viscous fluids were proposed. These are, in chronological order, the methods of Takahashi et al. [TDF*15], Peer et al. [PICT15, PT16], Bender and Koschier [BK17] and Barreiro et al. [BGFAO17]. Each of the approaches is able to simulate complex effects and has its own strengths and weaknesses.

In this work, we provide a theoretical in-depth analysis and discussion of the latest and most effective SPH-based implicit viscosity solvers proposed in the field of computer graphics. We present several quantitative and qualitative comparisons of the methods to investigate the quality and realism of the produced flow. From this analysis we formulated the following set of requirements that a physically consistent solver for highly-viscous flow should fulfill:

(R1) The method should be linear and angular momentum conserving. This also implies that the non-linearity in the velocity field and rigid body motions are conserved.
(R2) No spurious viscosity or numerical damping effects should be evident.
(R3) The solver should yield a physically meaningful viscous flow under spatial and temporal refinement, i.e. the viscous behavior should not change for different, reasonably fine spatial and temporal resolutions.
(R4) Complex physical effects that can be observed in real viscous fluids should be reproducible, e.g. buckling and rope coiling.

Moreover, the solver should ideally be computationally efficient and have a low memory consumption, although these are no strict requirements. As we will show in our detailed analyses, each of the mentioned solvers fails to meet at least one of the stated requirements. Based on the gained insights, we developed a novel implicit viscosity solver that is presented in this paper. Our method behaves well under spatial and temporal refinement and therefore supports adaptive time stepping. It is linear and angular momentum conserving and able to maintain rigid body motions considerably better than the existing state-of-the-art methods. By a combination with a divergence-free solver we ensure a divergence-free velocity field prior to the implicit viscosity solve in order to avoid any undesired bulk viscosity effects. The provided discrete Laplace operator moreover ensures that no spurious viscosity effects in poorly sampled regions, such as the free surface, occur. Finally, we demonstrate in several highly complex experiments that our novel approach produces more realistic results than state-of-the-art methods, show that our method has comparable performance as previous implicit approaches and that it has no additional memory requirements.

Besides the implicit viscosity solver, we present a sophisticated method for the simulation of melting objects. We model thermal conduction using the heat equation and induce material softening by placing a heat source in the simulation environment. Moreover, we discretize the heat equation to numerically simulate the temperature flow. We further generalize the viscosity model to a spatially varying viscosity field and couple the temperature changes with this field using the exponential model of Reynolds.

2. Related Work

This section is organized as follows. First, Eulerian approaches are briefly discussed. Second, methods based on the Lagrangian simulation methodology are reviewed. Finally, the most recent and closely related approaches for implicit handling of viscous forces in SPH discretizations are introduced.

Eulerian approaches Following the Newtonian constitutive model for fluids the viscous forces are linearly dependent on the divergence of the fluid's Cauchy strain rate. In case of an incompressible fluid these forces reduce to a weighted Laplacian of the velocity field. These two measures are used by the vast majority of approaches to simulate viscous fluids. In an early work of Foster and Metaxas [FM96] the Laplacian was discretized using finite differences for the simulation of incompressible fluids. However, they employed an explicit numerical integration scheme for viscous forces that limits the time step size in the presence of high viscosities. In order to improve the stability for larger time steps Stam [Sta99] developed an approach for stable implicit time updates. Later, Rasmussen et al. [REN*04] extended the formulation to model spatially-varying viscosity based on the strain rate tensor. While for incompressible fluids both measures are mathematically identical, the results of the according simulations can vary if the underlying velocity field is numerically not divergence-free. Therefore, Batty and Bridson [BB08] enforce a divergence-free velocity field in each solver substep. Their approach was later extended to support spatial adaptivity on unstructured meshes [BH11]. In a very recent work of Larionov et al. [LBB17] an implicit variational solver for highly-viscous liquids is proposed that combines the enforcement of incompressibility and the implicit computation of viscous forces into a single step.

Lagrangian approaches In the context of Lagrangian simulation methodologies several approaches to simulate viscous behavior were proposed in the past. Problems of a direct discretization of the Laplacian differential operator using the SPH formalism are the operator's sensitivity to particle disorder and sign changes of the second derivative of the usually employed kernels. Therefore, the usage of a direct SPH discretization of the Laplace operator results in an inconsistent behavior [Mon05]. To solve the problem, Brookshaw [Bro85] and Cleary and Monaghan [CM99] suggest to use a finite difference based integral approximation to get the first derivative before developing a Taylor series to obtain the second order derivative. This approach was also adopted by Morris and Monaghan [MM97]. Müller et al. [MCG03] designed a specific kernel function whose second derivative is positive on the entire kernel support domain to directly employ a non-defective discrete Laplacian operator. In order to avoid computing second order derivatives, an artificial viscosity model (XSPH) was employed by Monaghan [Mon92]. This strategy was later adopted in several works, e.g. [SB12, MM13, KB17]. As viscosity is assumed to occur due to friction within the fluid, the usage of generalized (non-Newtonian) constitutive models is also justified. In this regard, Paiva et al. [PPLT09] model viscosity using a non-Newtonian constitutive law but combine it with an unphysical viscosity correction and XSPH. The formulation was also adopted in the more recent work by de Souza Andrade et al. [dSP*14]. Although not

intended for the simulation of highly viscous fluids, Bender et al. [BKKW17] proposed a micropolar constitutive model to simulate turbulent flow. The formulation augments the dynamics of material particles by rotational movement and couples linear and angular momentum equation using a dissipative and a non-dissipative viscosity parameter.

The simulation of viscous fluids is also strongly related to the simulation of viscoelastic objects. An early particle-based approach for the simulation of viscoelasticity was proposed by Clavet et al. [CBP05]. While elasticity was enforced by connecting particles with springs, viscous behavior was simulated using an impulse-based velocity filter. Takahashi et al. [TNF14] employ a similar spring-based concept and simulate viscosity using an unphysical constraint model. A similar method for viscosity was proposed by Takahashi et al. [TDFN14] using a position-level constraint to derive a velocity filter. A point-based approach was presented by Jones et al. [JWJ*14] for the simulation of large viscoplastic deformations. In order to simulate viscous behavior they follow the approach of Müller et al. [MCG03] using a specialized viscosity kernel. Approaches for the simulation of viscous threads and sheets were proposed by Bergou et al. [BAV*10] and Batty et al. [BUAG12], respectively. Both methods derive a viscous potential from a known elastic potential by replacing strain with strain-rate. A mesh-based codimensional approach for thin sheets, narrow filaments, and small droplets was proposed by Zhu et al. [ZLQF15].

SPH based implicit viscosity solvers As the simulation of highly viscous fluids imposes strong restrictions on the time step size when using explicit time integration, implicit viscosity solvers for SPH discretizations have become popular in recent years. Takahashi et al. [TDF*15] model viscosity based on the strain-rate and use a backward Euler scheme for time integration. Peer et al. [PICT15] propose an implicit solver that decomposes the velocity gradient, projects the field onto a shear rate reduced state and reconstructs the velocity field. The method was later extended in [PT16] by adding vorticity diffusion to improve the target spin rate. Bender and Koschier [BK17] propose a constraint based formulation where they project the strain rate tensor onto a reduced state. Barreiro et al. [BGFAO17] introduce an approach for the simulation of viscoelastic materials. They integrate viscous forces in an implicit manner using a conformation constraint formulation with compliance factors based on XPBD [MMC16].

In this paper we propose a novel Lagrangian implicit viscosity solver for SPH discretizations of incompressible fluids. We show that all the closely related approaches discussed in the last paragraph suffer from physical inconsistencies such that they cannot fulfill all requirements (R1)-(R4). A detailed elaboration on the reasons for that is provided in Section 6. In contrast, our approach fulfills all requirements and is able to produce highly realistic flows.

3. Overview

The Navier-Stokes equations for incompressible fluids are derived by substituting the stress tensor of a fluid $\mathbf{P}$ into the momentum equation:

$$\rho \frac{D\mathbf{v}}{Dt} = \nabla \cdot \mathbf{P} + \mathbf{f}, \tag{1}$$

Algorithm 1 Simulation step

1: compute non-pressure forces $\mathbf{f}^{\text{np}}$
2: update velocity: $\mathbf{v}^{\text{np}} \leftarrow \mathbf{v}(t) + \Delta t \mathbf{f}^{\text{np}}/m$
3: adapt time step size Δt (CFL condition)
4: $\mathbf{v}^{\text{cd}} \leftarrow$ constantDensitySolver($\mathbf{v}^{\text{np}}$)
5: update position: $\mathbf{x}(t+\Delta t) \leftarrow \mathbf{x}(t) + \Delta t \mathbf{v}^{\text{cd}}$
6: $\mathbf{v}^{\text{df}} \leftarrow$ divergenceFreeSolver($\mathbf{v}^{\text{cd}}$)
7: $\mathbf{v}(t+\Delta t) \leftarrow$ implicitViscositySolver($\mathbf{v}^{\text{df}}$)

where ρ, $\mathbf{v}$ and $\mathbf{f}$ are density, velocity and body forces per unit volume. The stress tensor of a Newtonian fluid is defined as

$$\mathbf{P} = -p\mathbf{I} + \mu\left(\nabla\mathbf{v} + (\nabla\mathbf{v})^T\right), \tag{2}$$

where p, μ and $\mathbf{I}$ are pressure, dynamic viscosity and the identity matrix. For incompressible fluids the continuity equation $\frac{D\rho}{Dt} = -\rho\nabla\cdot\mathbf{v} = 0$ must be satisfied which yields that the velocity field of the fluid must be divergence-free, i.e. $\nabla\cdot\mathbf{v} = 0$. Finally, the equation of motion for a divergence-free velocity field is defined as

$$\rho\frac{D\mathbf{v}}{Dt} = -\nabla p + \mu\underbrace{\nabla\cdot\nabla\mathbf{v}}_{\nabla^2\mathbf{v}} + \mu\underbrace{\nabla\cdot(\nabla\mathbf{v})^T}_{\nabla(\nabla\cdot\mathbf{v})=\mathbf{0}} + \mathbf{f} \tag{3}$$

$$\frac{D\mathbf{v}}{Dt} = -\frac{1}{\rho}\nabla p + \nu\nabla^2\mathbf{v} + \frac{\mathbf{f}}{\rho}, \tag{4}$$

where $\nu = \frac{\mu}{\rho}$ denotes the kinematic viscosity.

The stress tensor in Equation (2) consists of a pressure term $-p\mathbf{I}$ and a viscosity term based on the strain rate tensor

$$\mathbf{E} = \frac{1}{2}(\nabla\mathbf{v} + (\nabla\mathbf{v})^T). \tag{5}$$

Therefore, recent implicit viscosity solvers [TDF*15, PICT15, PT16, BK17, BGFAO17] use a strain rate based formulation to compute viscous forces. However, such a formulation has two problems. First, most current pressure solvers, like IISPH [ICS*14] or PBF [MM13], do not enforce a divergence-free velocity field, but a non-zero divergence implies that $\nabla\cdot(\nabla\mathbf{v})^T \neq \mathbf{0}$ (cf. Equation (3)) and therefore leads to undesired bulk viscosity [Lau11, PICT15]. The second, more critical problem of using a strain rate based formulation became apparent when we performed an extensive analysis of the most important SPH concepts to simulate highly viscous materials. In this analysis we found out that the SPH strain rate computation used in all recent solvers yields a significant error at the free surface due to particle deficiency (see Section 6). This error leads to ghost forces in the simulation which cause severe visual artifacts and a significant loss of angular momentum as we will show in Section 7.

In our work we introduce an implicit viscosity solver which is directly based on the Laplacian of the velocity field instead of using the strain rate (see Section 4). We propose to determine the Laplacian using a combination of SPH and finite differences to get a viscosity that is Galilean invariant, vanishes for rigid body rotation and conserves linear and angular momentum. In this way we are able to avoid any ghost forces at the free surface. Moreover, we combine our viscosity solver with the pressure solver DFSPH [BK17] which enforces a divergence-free velocity field. This avoids unde-

sired bulk viscosity. Finally, we present an extension of our viscosity formulation to simulate melting effects (see Section 5).

Algorithm 1 outlines a simulation step with our novel approach. First, we compute intermediate velocities $\mathbf{v}^{\text{np}}$ only considering non-pressure forces (e.g. gravity) except viscosity. In the next step we adapt the time step size according to the Courant-Friedrich-Levy (CFL) condition [Mon92, IOS*14]:

$$\Delta t \leq 0.4 \frac{d}{\|\mathbf{v}_{\max}\|},$$

where d is the particle diameter and $\mathbf{v}_{\max}$ is the maximum particle velocity. Then pressure forces are determined and the particle positions are updated to enforce a constant density. A divergence-free velocity field is enforced in the subsequent step. This divergence-free field is the input of our implicit viscosity solver which computes the final velocities $\mathbf{v}(t+\Delta t)$ (see Section 4).

4. Implicit Viscosity

In this section we introduce a novel efficient implicit viscosity solver that is momentum conserving, behaves well under spatial and temporal refinement and requires no additional memory. Moreover, our method avoids ghost forces emerging in strain rate based formulations by directly using the Laplacian of the velocity field and undesired bulk viscosity by enforcing a divergence-free velocity field using DFSPH.

The Navier-Stokes equations for incompressible fluids (see Section 3) define the viscous force of a fluid as $\mathbf{f}^{\text{v}}(\mathbf{v}) = \mu\nabla^2\mathbf{v}$. The standard SPH discretization of the Laplacian

$$\nabla^2\mathbf{v}_i = \sum_j \frac{m_j}{\rho_j}\mathbf{v}_j\nabla^2 W_{ij} \tag{6}$$

is, however, problematic since it uses the second derivative of the kernel function, which for Gaussian-like kernels changes the sign inside its support radius. Therefore, the computation is prone to noise in the particle distribution [Pri12]. There exist two common solutions for this problem. First, the second derivative can be determined by taking two first derivatives [FMH*94, WBF*96, TDF*15]. However, this approach introduces additional smoothing and increases memory consumption as well as computation time. Second, a combination of an SPH first derivative with finite differences can be used to compute the Laplacian. This idea was first proposed by Brookshaw [Bro85] and has later been used in numerous works, both for scalar quantities [Mon92, CM99, IOS*14] as well as for vector quantities [ER03, JSD04, Mon05, Pri12].

We adopt such a combination of SPH derivative and finite differences to compute the Laplacian of the velocity field [Mon05]:

$$\nabla^2\mathbf{v}_i = 2(d+2)\sum_j \frac{m_j}{\rho_j}\frac{\mathbf{v}_{ij}\cdot\mathbf{x}_{ij}}{\|\mathbf{x}_{ij}\|^2+0.01h^2}\nabla W_{ij}, \tag{7}$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$ and d is the number of spatial dimensions. The term $0.01h^2$ is required to prevent singularities. Equation (7) has the advantage that it is Galilean invariant, vanishes for rigid body rotation and conserves linear and angular momentum [Mon92].

Implicit time integration For our time integration we use the splitting concept to decouple the pressure solve and the viscosity solve, which is common practice in SPH [IOS*14]. This allows us to use efficient methods for both steps. For the integration of the viscous forces $\mathbf{f}^{\text{v}}(\mathbf{v})$ we use the implicit discretization

$$\mathbf{v}(t+\Delta t) = \mathbf{v}^{\text{df}} + \Delta t\frac{\mu}{\rho}\nabla^2\mathbf{v}(t+\Delta t). \tag{8}$$

This system is linear and can also be written as

$$(\mathbf{I} - \Delta t\mathbf{A})\mathbf{v}(t+\Delta t) = \mathbf{v}^{\text{df}}, \tag{9}$$

where the matrix $\mathbf{A}$ consists of 3×3 blocks

$$\mathbf{A}_{ij} = -2(d+2)\frac{\mu\bar{m}_{ij}}{\rho_i\rho_j}\frac{\nabla W_{ij}\mathbf{x}_{ij}^T}{\|\mathbf{x}_{ij}\|^2+0.01h^2}, \qquad \mathbf{A}_{ii} = -\sum_j \mathbf{A}_{ij}. \tag{10}$$

Here $\mathbf{A}_{ij}$ denotes the matrix block that corresponds to particles i and j. We use the average mass $\bar{m}_{ij} = (m_i + m_j)/2$ to get a symmetric and positive definite linear system. Hence, we can solve it using the conjugate gradient method. To save both computation time and memory, we use a matrix-free implementation. Moreover, we apply a block version of the Jacobi preconditioner to improve the convergence of our solver. This preconditioner consists of the diagonal 3×3 blocks $\mathbf{I} - \Delta t\mathbf{A}_{ii}$ of the matrix. Even though inverting the blocks slightly increases computation time in comparison to a standard Jacobi preconditioner, the block version showed a better improvement of the convergence and the performance in our experiments. Moreover, we started the conjugate gradient solver with an initial guess of the solution to reduce the number of iterations. We propose to use $\mathbf{v}^{\text{df}} + \Delta\mathbf{v}$ as initial guess, where $\Delta\mathbf{v} = \mathbf{v}(t) - \mathbf{v}^{\text{df}}(t-\Delta t)$ is the velocity difference determined by the viscosity solver in the last step.

Boundary handling We adopt the rigid-fluid coupling of Akinci et al. [AIA*12] to simulate interaction with solid boundaries. For each boundary particle i a pseudo-mass is computed as $\Psi_i = \frac{\rho_0}{\sum_k W_{ik}}$, where k denotes the indices of neighboring boundary particles. These pseudo-masses are used in both the pressure solve and the viscosity solve. To control the boundary's stickiness we define a boundary viscosity coefficient μ_b. Finally, the influence of the boundary particles is considered by extending the diagonal blocks and adapting the right hand side $\mathbf{b}$ of the linear system:

$$\mathbf{A}_{ii} = -\sum_j \mathbf{A}_{ij} + 2(d+2)\sum_k \frac{\mu_b\Psi_k}{\rho_i^2}\frac{\nabla W_{ik}\mathbf{x}_{ik}^T}{\|\mathbf{x}_{ik}\|^2+0.01h^2}, \tag{11}$$

$$\mathbf{b}_i = \mathbf{v}_i^{\text{df}} - 2(d+2)\Delta t\sum_k \frac{\mu_b\Psi_k}{\rho_i^2}\frac{\mathbf{v}_k\cdot\mathbf{x}_{ik}}{\|\mathbf{x}_{ik}\|^2+0.01h^2}\nabla W_{ik}. \tag{12}$$

The preconditioner must be adapted accordingly. To get a consistent system, a reaction force acting on the boundary particles is determined according to the method of Akinci et al. [AIA*12]. Thanks to this extension of the linear system, we can efficiently simulate sticky and separating boundaries using a conjugate gradient solver while Peer et al. [PICT15, PT16] fall back to a Jacobi solver in the latter case.

5. Temperature-Dependent Viscosity

In this section we extend our viscosity solver for the simulation of melting effects. Melting is an interesting phenomenon that strongly

relates temperature to viscosity. For that purpose we need to keep track of the fluid temperature T which changes according to

$$\frac{\partial T}{\partial t} = \frac{\partial T^{\mathrm{r}}}{\partial t} + \frac{\partial T^{\mathrm{t}}}{\partial t} + \frac{\partial T^{\mathrm{f}}}{\partial t}, \tag{13}$$

where $\frac{\partial T^{\mathrm{r}}}{\partial t}$ is the heat radiated to and from the surrounding, $\frac{\partial T^{\mathrm{t}}}{\partial t}$ captures heat transfer from the enclosing fluid and $\frac{\partial T^{\mathrm{f}}}{\partial t}$ models the influx of thermal energy from heat sources. Surface particles try to match a given room temperature T^{room} and radiate heat at a rate of

$$\frac{\partial T^{\mathrm{r}}}{\partial t} = \frac{T^{\mathrm{room}} - T}{t^{\mathrm{r}}}, \tag{14}$$

where t^{r} is the radiation half time. We use the magnitude of the surface normal to detect which particles lie at the surface [IOS*14]. The heat transfer inside the fluid is determined by the heat equation

$$\frac{\partial T^{\mathrm{t}}}{\partial t} = -\frac{k}{\rho c}\nabla^2 T, \tag{15}$$

where k is the thermal conductivity and c denotes the specific thermal capacity. The scalar Laplacian is computed as [Bro85]

$$\nabla^2 T_i = 2\sum_j \frac{m_j}{\rho_j}\left(T_i - T_j\right)\frac{\mathbf{x}_{ij}\cdot\nabla W_{ij}}{\left\|\mathbf{x}_{ij}\right\|^2 + 0.01h^2}. \tag{16}$$

Our model supports spherical and directional heat sources. A spherical heat source has an inverse square falloff. The heat flux at distance d from the source is

$$Q(d) = W^0 \cdot \left(\frac{\sqrt{2}-1}{r}d + 1\right)^{-2}, \tag{17}$$

where W^0 and r are the wattage and the radius of the source, respectively. r scales the flux such that $Q(r) = \frac{1}{2}Q(0)$, which is equivalent to having a spherical body of radius r with a flux of $\frac{1}{2}Q(0)$ at its surface. The energy arriving at a particle located at $\mathbf{x}$ is

$$\frac{\partial T^{\mathrm{f}}}{\partial t} = AQ(d_s(\mathbf{x})), \tag{18}$$

where A is the cross-sectional area of the particle and $d_s(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_s\|$ measures the distance to the source center $\mathbf{x}_s$. For a directional heat source we replace the distance function with $d_d(\mathbf{x}) = \|\mathbf{n}_d^T(\mathbf{x} - \mathbf{x}_d)\|$. Here $\mathbf{n}_d$ is the source direction and $\mathbf{x}_d$ is a point on a virtual plane with normal $\mathbf{n}_d$ on which the flux has a value of $Q(0)$. We apply the influence of heat sources only to surface particles.

Since temperature changes are, in general, a slow process, we use explicit time integration for the heat update: $T(t+\Delta t) = T(t) + \Delta t \frac{\partial T}{\partial t}(t)$. After updating particle temperatures, the viscosity values are computed using Reynolds' exponential model $\mu = \mu^0 e^{\alpha(T^0 - T)}$, where μ^0 is the viscosity at temperature T^0 and α is a positive scaling factor. The resulting fluid has varying viscosities for each particle. To keep our system symmetric, we average the coefficient μ between a particle and its respecting neighbor in Equation (10).

6. Discussion

In this section we will first introduce the concepts of the most important implicit SPH viscosity solvers in chronological order and analyze their advantages and disadvantages. Then we show that all previous works use a formulation based on the strain rate of the velocity field and that the SPH strain rate computation yields a large error at the free surface due to particle deficiency. This error causes visual artifacts and a significant loss of angular momentum. Finally, we show that our novel implicit method solves this problem.

Takahashi et al. [TDF*15] The authors propose an implicit integration scheme for a stable simulation of highly viscous fluids. They determine the second derivative of the velocity field by taking two first derivatives. First, the strain rate $\mathbf{E}_i$ is computed for all particles using Equation (5) and then the divergence is determined:

$$\nabla\cdot\left(\nabla\mathbf{v}_i + (\nabla\mathbf{v}_i)^T\right) = \sum_j m_j\left(\frac{2\mathbf{E}_i}{\rho_i^2} + \frac{2\mathbf{E}_j}{\rho_j^2}\right)\nabla W_{ij}. \tag{19}$$

Finally, the implicit integration step is performed by solving

$$\mathbf{v}(t+\Delta t) = \mathbf{v}^{\mathrm{np}} + \frac{\Delta t}{\rho}\mu\nabla\cdot\left(\nabla\mathbf{v}(t+\Delta t) + (\nabla\mathbf{v}(t+\Delta t))^T\right). \tag{20}$$

Takahashi et al. propose an implicit integration for viscous fluids that ensures that the viscosity is independent of the time step size and the spatial resolution. However, determining the second derivative by computing two first derivatives with SPH requires to consider all second-ring neighbors. This increases the complexity of the solver significantly and results in a low performance (see Section 7). Another issue of this method is that a divergence-free velocity field is not enforced but the divergence of the strain rate is used to compute the viscous force. For a velocity field that is not divergence-free this causes undesired bulk viscosity [PICT15].

Peer et al. [PICT15, PT16] The core idea of Peer et al. is to decompose the velocity gradient into three components:

$$\nabla\mathbf{v} = \mathbf{R} + \mathbf{V} + \mathbf{S}, \tag{21}$$

where $\mathbf{R} = \frac{1}{2}(\nabla\mathbf{v} - (\nabla\mathbf{v})^T)$ is the spin rate tensor, $\mathbf{V} = \frac{1}{3}(\nabla\cdot\mathbf{v})\mathbf{I}$ the expansion rate tensor and $\mathbf{S} = \mathbf{E} - \mathbf{V}$ the traceless shear rate tensor (cf. Equation (5)). Then the shear rate in Equation (21) is reduced by a factor $0 \le \xi \le 1$ to obtain a target velocity gradient $\nabla\mathbf{v}^t = \mathbf{R} + \mathbf{V} + \xi\mathbf{S}$. Finally, the velocity field of the fluid is reconstructed by a first-order Taylor approximation

$$\mathbf{v}_i(t+\Delta t) = \sum_j \frac{m_j}{\rho_j}\left(\mathbf{v}_j(t+\Delta t) + \frac{\nabla\mathbf{v}_i^t + \nabla\mathbf{v}_j^t}{2}\mathbf{x}_{ij}\right)W_{ij}. \tag{22}$$

The resulting linear system can be solved efficiently using a conjugate gradient method. Peer and Teschner [PT16] extend this approach by adding vorticity diffusion, where another linear system is solved to get an improved target spin rate tensor.

The main disadvantage of both methods is that the reconstruction of the velocity field in Equation (22) introduces a significant damping (see Section 7) due to the fact that SPH can not reconstruct linear fields as discussed in [BGFAO17]. Moreover, the approach cannot simulate low viscous fluids realistically. Another limitations is that the parameter ξ depends on the time step size and the spatial resolution and is not physically meaningful.

Bender and Koschier [BK17] In the work of Bender and Koschier first the strain rate $\mathbf{E}$ is determined and then a constraint $\mathbf{C}_i(\mathbf{v}) = \mathbf{E}_i - \gamma\mathbf{E}_i$ is defined for each particle, where $0 \leq \gamma \leq 1$ is a stiffness parameter. Since the strain rate tensor is symmetric, the authors write the constraints as six-dimensional vector functions and compute a Lagrange multiplier for each constraint using a Jacobi solver. To compute this Lagrange multiplier in each simulation step a six-dimensional matrix has to be inverted for each constraint.

Bender and Koschier combine their viscosity solver with DFSPH to get a divergence-free velocity field and to avoid bulk viscosity. In contrast to Peer et al. this approach is also able to handle low viscous fluids. However, their stiffness parameter γ is also time step and resolution dependent and not physically meaningful. The usage of a Jacobi solver and the requirement to invert a six-dimensional matrix for each constraint causes a large computational effort which results in poor performance.

Barreiro et al. [BGFAO17] Recently, Barreiro et al. introduced a method to simulate viscoelastic polymer fluids. The authors define six-dimensional velocity and position constraints for the symmetric polymer conformation tensor. The constraints depend on the strain rate $\mathbf{E}$ of the fluid to simulate viscous behavior. Their implicit constraint solver is based on XPBD [MMC16] to ensure a physically meaningful behavior under temporal and spatial refinement.

This method supports different material behaviors that range between elastoplastic, inviscid and highly viscous fluids. However, the position-based pressure solver does not enforce a divergence-free velocity field, which leads to undesired bulk viscosity. Furthermore, the linear system is solved by Jacobi iteration, which converges slowly. Finally, the material parameters are only physically meaningful for polymer fluids. Other viscoelastic materials have to be modeled artistically.

Comparison with our method The approaches of Peer et al. [PICT15, PT16] and Bender and Koschier [BK17] enforce a target velocity gradient and a target strain rate, respectively. While being efficient, this results in a viscous behavior that is dependent on the time step size and the spatial resolution. Moreover, the viscosity coefficients of both methods are not physically meaningful. Our approach and the method of Takahashi et al. [TDF*15] avoid these problems by using an implicit time integration. However, Takahashi et al. require the second-ring neighbors in their linear system, which causes a significant computational effort, while our approach only requires the one-ring neighborhood and is therefore considerably more efficient as discussed in Section 7. The method of Barreiro et al. [BGFAO17] solves compliant constraints using a Jacobi solver and does therefore also ensure a physically meaningful behavior under spatial and temporal refinement.

In contrast to our approach all five methods introduced above use a strain rate based formulation, where the strain rate (see Equation (5)) is determined using the SPH formulation for a symmetric velocity gradient [Mon92]:

$$\nabla \mathbf{v}_i = \frac{1}{\rho_i} \sum_j m_j (\mathbf{v}_j - \mathbf{v}_i) \nabla W_{ij}^T. \tag{23}$$

However, this computation of the strain rate tensor is negatively affected by particle deficiency at the free surface which we demonstrate in a simple experiment. Figure 2 (left) shows the 2D velocity field of 21×21 particles rotating around the origin. For each particle we determined the strain rate tensor using the velocity gradient in Equation (23). Figure 2 (right) depicts the Frobenius norm of the resulting tensors which should be zero for a pure rotation. The figure demonstrates that the strain rate computation yields a considerable error at the free surface. This error leads to ghost forces and therefore causes visual artifacts and a loss of angular momentum as we will demonstrate in Section 7.

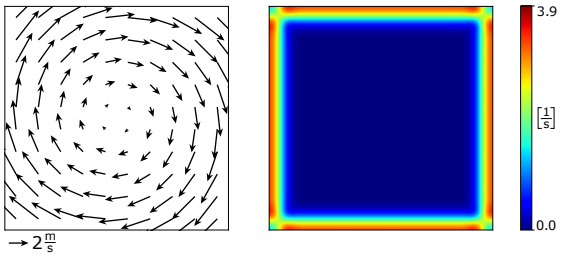


Figure 2: Particle deficiency problem at the free surface when computing the strain rate. Left: 2D velocity field of 21×21 particles rotating around the origin in a unit box. Right: Frobenius norm of the corresponding strain rate tensors determined using Equation (23).

Viscosity solver	(R1)	(R2)	(R3)	(R4)
Peer et al. [PICT15]	✗	✗	✗	(✓)
Peer and Teschner [PT16]	✗	✗	✗	(✓)
Bender and Koschier [BK17]	✗	✓	✗	(✓)
Takahashi et al. [TDF*15]	✗	✗	✓	(✓)
Barreiro et al. [BGFAO17]	(✓)	✗	✓	(✓)
Our approach	✓	✓	✓	✓

Table 1: The previous methods do not satisfy all of the requirements defined in Section 1.

Peer et al. [PICT15, PT16], Bender and Koschier [BK17] and Barreiro et al. [BGFAO17] directly use the resulting strain rate tensors in their formulation while Takahashi et al. [TDF*15] compute the divergence of the strain rate to get a viscous force. In our work we determine the viscous force using the Laplacian of the velocity field. In theory for a divergence-free velocity field both forces should be exactly the same since in this case $\nabla \cdot (\nabla \mathbf{v} + \nabla(\mathbf{v})^T) = \nabla^2 \mathbf{v}$ (cf. Equation (3)). However, the method proposed by Takahashi et al. has two issues. First, they use IISPH as pressure solver which does not enforce a divergence-free velocity field. This leads to undesired bulk viscosity. Second, their viscous force is based on the strain rate computation which has problems at the free surface as discussed above. Figure 3 compares the approach of Takahashi et al. with our method. The error in the left figure shows that the method of Takahashi et al. generates erroneous forces at the free surface while the maximal error of our approach (right) was below $10^{-12} \frac{1}{\text{m}\cdot\text{s}}$. The reason for the low error is that our formulation of the Laplacian is Galilean invariant, vanishes for rigid body rotation and conserves linear and angular momentum.

Table 1 summarizes which of the requirements defined in Sec-

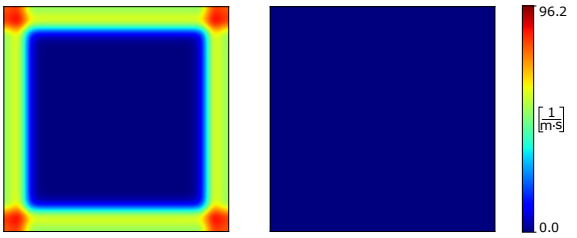


Figure 3: Comparison of Takahashi et al. with our method using a rotating velocity field (see Figure 2 (left)). Left: $\|\nabla \cdot (\nabla \mathbf{v} + \nabla(\mathbf{v})^T)\|$ determined by the formulation of Takahashi et al., Right: Laplacian of the velocity field $\|\nabla^2 \mathbf{v}\|$ that is used in our work.

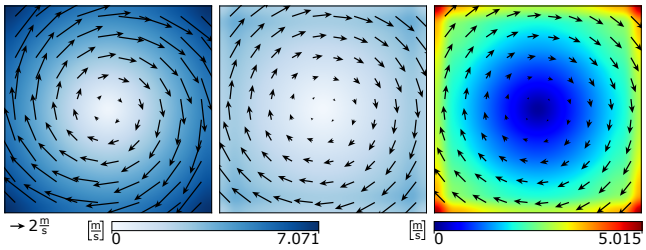


Figure 4: Reconstruction error of Peer et al. [PICT15, PT16]. Left: Velocity field of particles rotating around the center. Middle: Reconstructed velocity field using the first-order Taylor approximation. Right: The difference between the left and the middle plot shows the motion that gets lost due to the reconstruction of the velocity field using SPH.

tion 1 are satisfied by the discussed methods. The strain rate error due to particle deficiency at the free surface leads to ghost forces which affect the angular momentum conservation considerably. This is also shown in Section 7. Hence, none of the former methods fulfills the first requirement. However, Barreiro et al. improve the conservation of angular momentum significantly by using a corotational constitutive model to account for the nonlinearity in the position-level motion. The second requirement is only met by Bender and Koschier and our novel method since all other approaches either generate bulk viscosity by ignoring the velocity divergence, or numerical damping due to the fact that SPH cannot reconstruct linear fields as discussed in [BGFAO17]. The methods of Peer et al. and Bender and Koschier do not behave well under spatial and temporal refinement. All approaches are able to simulate complex physical effects. However, since none of the previous methods satisfies all requirements, their results are less realistic.

7. Results

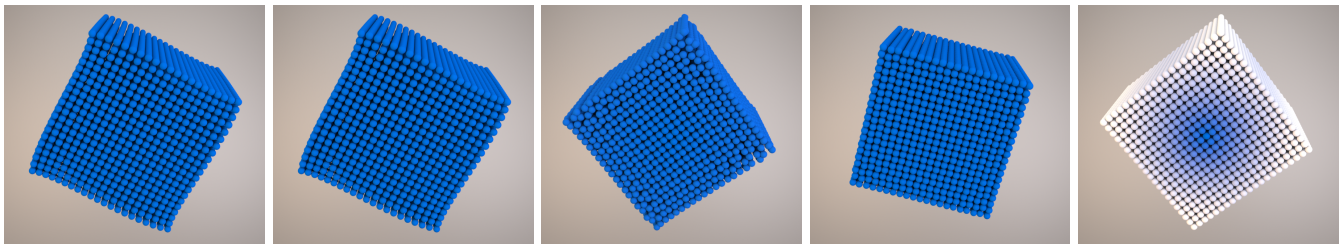
In this section we make different comparisons with four of the implicit viscosity methods [TDF*15, PICT15, PT16, BK17] which were discussed in the last section. Since the work of Barreiro et al. [BGFAO17] was published just recently, we were not able to implement their method for a practical comparison. In the experiments we demonstrate that the ghost forces which occur in previous approaches lead to a considerable loss of angular momentum and to noticeable artifacts. Moreover, we show that our novel approach solves these problems.

For the comparisons in this section we used the open-source framework SPlisHSPlasH [Ben17], which implements the methods of Peer et al. [PICT15], Peer and Teschner [PT16] and Bender and Koschier [BK17], and we additionally implemented the method of Takahashi et al. [TDF*15]. In all comparisons we used a fixed time step size since not all of these methods behave consistently for varying time step sizes. Unless stated otherwise, boundary viscosity was chosen to be the same as the fluid viscosity.

Momentum conservation First, we compare the momentum conservation of all approaches. As already mentioned in the last section the methods of Peer et al. and Peer and Teschner introduce a significant damping due to their reconstruction of the velocity field. We demonstrate this by performing one reconstruction step using Equation (22) for a 2D velocity field of particles rotating around the origin. Figure 4 shows the field before the reconstruction on the left and we can see that the motion is considerably damped after the reconstruction in the middle. On the right is a plot of the motion that gets lost. In this experiment more than 70% of the velocities were damped out in just one step since SPH cannot reconstruct linear fields [BGFAO17]. This demonstrates that both methods are far from being momentum conserving.

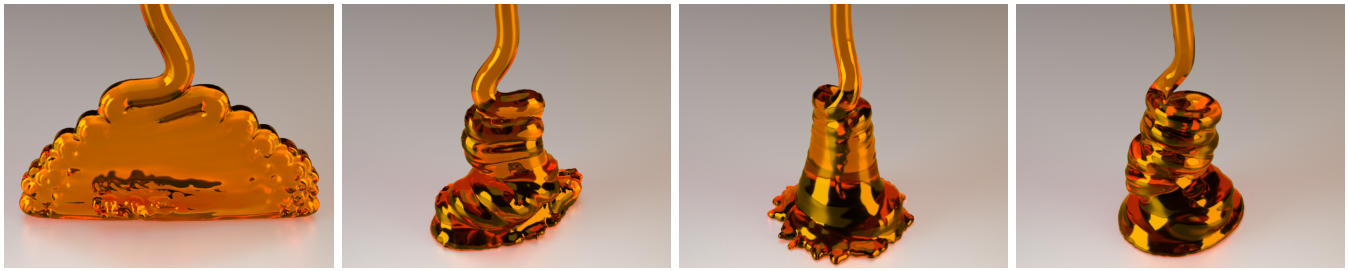
In another experiment we simulated a highly viscous fluid cube with all methods. At the beginning of the experiment the cube has a rotational motion. After less than 0.5s the cube simulated with our method is the only one which is still rotating and which is not artificially damped (see Figure 5). This can also be seen in the accompanying video. The reason for this loss of angular momentum in all previous approaches is the strain rate error at the free surface of the cube (see Section 6). Due to this error the methods compute viscous forces at the free surface. These ghost forces counteract the rotational motion. Peer et al. [PICT15, PT16] damp the rotational motion even more by their reconstruction of the velocity field. Since our approach is based on a Laplacian that vanishes for rigid body motion and that conserves linear and angular momentum, the rotation of the cube is not negatively influenced.

Complex physical effects To compare the different methods with respect to realism, we simulated the rope coiling effect (see Figure 6). This effect is well-known in physics and one important insight in this area is that the coil radius is roughly constant when the gravitational force is dominant [HMG*06]. Hence, this is an important detail for realistic results. Since the method of Peer and Teschner [PT16] is an improved version of their older approach [PICT15], we only considered the newer method. In the simulation with this method the rotational motion got lost due to the damping introduced by their reconstruction of the velocity field. Therefore, only a buckling motion occurred. With the approach of Bender and Koschier [BK17] a slow coiling is visible but the rotational motion is also damped due to the strain rate error at the free surface as discussed above. Using the method of Takahashi et al. [TDF*15] coiling gets faster during the simulation since the coil radius gets smaller. This can also be seen in the original video of the authors. We assume that this comes from the strain rate error at the surface and the spurious bulk viscosity. In contrast, our ap-



(a) Peer et al. [PICT15] (b) Peer & Teschner [PT16] (c) Bender & Koschier [BK17] (d) Takahashi et al. [TDF*15] (e) Our method

Figure 5: Rotating cubes with color-coded velocities. The strain-rate based formulation of previous methods leads to ghost forces which stop the rotational motion after less than 0.5s while our approach conserves the angular momentum.



(a) Peer & Teschner [PT16] (b) Bender & Koschier [BK17] (c) Takahashi et al. [TDF*15] (d) Our method

Figure 6: Rope coiling effect simulated with different viscosity solvers.

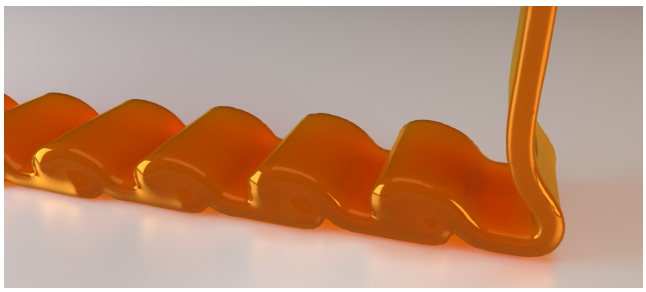


Figure 7: Simulation of the buckling effect with our approach.

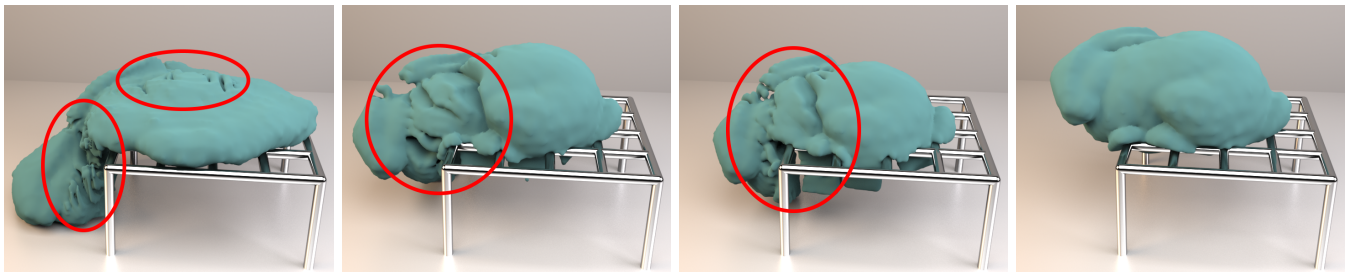
proach generates a realistic coiling motion where the coil radius stays almost constant.

Our method is also able to produce realistic buckling effects (see Figure 7). In this simulation a fluid emitter moved from left to right to generate a uniform buckling. Finally, Figure 1 (left) shows our boundary handling in a complex scenario, where a highly viscous fluid (406k particles) interacts with a fast rotating dough hook.

Quality In another experiment we compared the quality of the simulation results of the different approaches. In this experiment we dropped a bunny model with 56k fluid particles on a regular grid (see Figure 8). We set the boundary viscosity to zero for all methods to make the comparison independent of boundary handling. In the simulations with the methods of Peer and Teschner [PT16], Bender and Koschier [BK17] and Takahashi et al. [TDF*15] the surface of the bunny breaks up which causes severe artifacts. Similar artifacts at the surface can also be noticed in the original videos of all authors. The reason for these artifacts is the generation of ghost forces due to the particle deficiency problem in the computation of the strain rate, which we already discussed above. Moreover, the results show that the method of Peer and Teschner [PT16] introduced a significant numerical damping due to the reconstruction of the velocity field. In contrast, our approach provides a more realistic motion with no artifacts at the surface and no numerical damping.

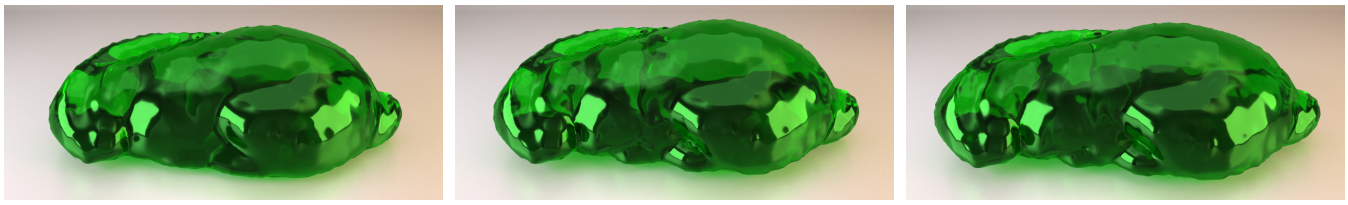
Note that the methods of Peer and Teschner [PT16] and Bender and Koschier [BK17] use viscosity parameters which are not physically meaningful. Therefore, in this experiment we tried to find some viscosity parameters so that the viscosity behavior in all simulations is approximately comparable. However, also note that choosing different viscosity parameters will not avoid the artifacts.

Temporal and spatial refinement In order to demonstrate that our method behaves well under temporal refinement, we simulated a viscous bunny with 56k particles using three different time step sizes: $\Delta t = 5$ms, $\Delta t = 0.5$ms and $\Delta t = 0.05$ms. This important property of our discretization allows us to use adaptive time stepping while maintaining a consistent viscous behavior. The results in Figure 9 and in the accompanying video show that in general the viscous behavior in all three simulations is the same. Of course there are small differences since a simulation with a smaller time step size generates more accurate results. In a second experiment we also demonstrate a consistent behavior under spatial refinement using three different resolutions (see Figure 10).



(a) Peer & Teschner [PT16] (b) Bender & Koschier [BK17] (c) Takahashi et al. [TDF*15] (d) Our method

Figure 8: Bunny model with 56k particles dropped on a regular grid. Previous methods show severe artifacts due to the particle deficiency problem of the strain rate computation. Our method solves this problem and provides more realistic results without artifacts.



(a) $\Delta t = 5$ms (b) $\Delta t = 0.5$ms (c) $\Delta t = 0.05$ms

Figure 9: Our method maintains the viscous behavior in a simulation with three different time step sizes but the same viscosity coefficient.

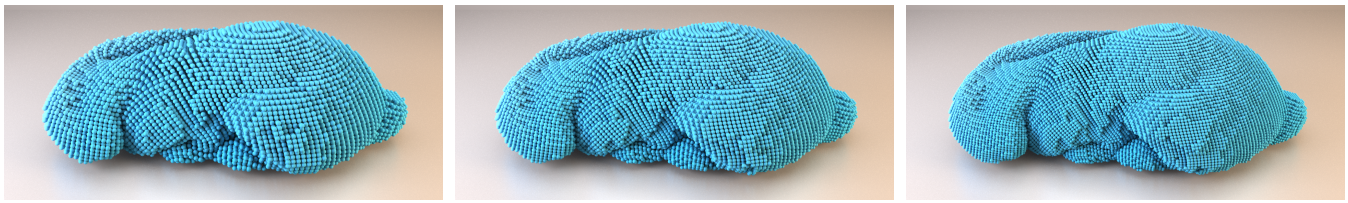
Performance As in previous works [TDF*15, PT16] we analyze the linear systems of all methods to compare the performance. We also adopt their assumption that a particle has about 30 first-ring and up to 320 second-ring neighbors.

Peer et al. [PICT15] use the conjugate gradient method to solve three systems of size $N \times N$, where N is the number of particles, while their improved approach [PT16] requires six systems of the same size. For their computation they only require the first-ring neighbors. Therefore, the complexity of each solver iteration is $3N \cdot 30 = 90N$ and $6N \cdot 30 = 180N$, respectively. However, note that in the case of separating boundaries Peer et al. cannot solve their system with the conjugate gradient method due to their divergence handling and fall back to a Jacobi solver. The method of Bender and Koschier [BK17] is also based on the first-ring neighborhood. They solve a system with N six-dimensional constraints resulting in a complexity of $6N \cdot 6 \cdot 30 = 1080N$ per iteration. However, instead of a conjugate gradient solver they use a Jacobi method which converges slower. Takahashi et al. [TDF*15] solve one $3N \times 3N$ system using the conjugate gradient method. But in their formulation they need the second-ring neighbors which yields a complexity of $3N \cdot 3 \cdot 320 = 2880N$ in each solver iteration. Finally, in our work we also use the conjugate gradient method to solve a $3N \times 3N$ system which only depends on the first-ring neighborhood. This leads to a complexity $3N \cdot 3 \cdot 30 = 270N$ per iteration.

Our approach reduces the complexity by more than a factor of ten compared to the method of Takahashi et al. while maintaining a physically meaningful behavior under spatial and temporal refinement. It has a higher complexity than the method of Peer et al. However, in case of separating boundaries the performance of the approach of Peer et al. is significantly reduced since in this case they fall back to a Jacobi solver due to their special divergence handling. Furthermore, in contrast to the approach of Peer et al., our method avoids the strain rate error at the free surface, is momentum conserving and ensures a physically consistent behavior.

After this theoretical performance analysis we also want to provide some performance measurements for our method. The performance was first measured in a simulation of a viscous statue (see Figure 1, right) with 1.1M particles. The boundary viscosity in this scene was set to zero. A simulation step on two Intel Xeon E5-2697 processors with 2.3GHz and 18 cores required 2.2s on average, of that the viscosity solve needed 1.4s, the neighborhood search 0.4s and the pressure solve 0.4s. We required an average of 10.3 iterations with the proposed initial guess and our preconditioner. Our block preconditioner reduced the number of iterations by a factor of two and the initial guess by a factor of eight which yields a significant speedup of the simulation. All values were measured in the first four seconds of the simulation where the statue falls down and interacts with the boundary. Further performance measurements are shown in Table 2.

Memory consumption Peer et al. [PICT15] require nine scalar values per particle to store the target velocity gradient. Their improved method [PT16] additionally needs memory for three values per particle for the vorticity. Bender and Koschier [BK17] store the target strain rate, a factor to compute a Lagrange multiplier and the multiplier itself. Therefore, they need 48 scalar values per particle. Takahashi et al. [TDF*15] keep the complete coefficient matrix in memory. In their work, they mention that they preserve 12GB memory for 500k fluid particles which corresponds to 3000 scalar values per particle. In contrast to previous approaches, our novel implicit



(a) 56k particles (b) 100k particles (c) 200k particles

Figure 10: Our method shows a consistent behavior in a simulation with three different particle resolutions but the same viscosity coefficient.

Model	Δt	**# particles**	$\mu\,[\frac{kg}{m \cdot s}]$	**neighb. search**	**pressure solver**	**viscosity solver**	**# viscosity iterations**	**total**
Dough hook (Figure 1, left)	1.6 (CFL)	405k	5000	67.9	281.6	762.8	16.9	1112.3
Statue (Figure 1, right)	0.5 (CFL)	1.1M	20000	405.7	419.3	1363.8	10.3	2188.8
Coiling (Figure 6)	1.0	25k	4000	8.9	6.4	28.2	15.9	43.5
Buckling (Figure 7)	1.0	80k	2000	19.2	20.3	103.2	16.7	142.7
Grid model (Figure 8)	1.0	56k	15000	28.6	22.0	73.2	10.4	123.8
Bunny model (Figure 9a)	5.0	56k	10000	16.1	36.8	131.6	23.8	184.5
Bunny model (Figure 9b & 10a)	0.5	56k	10000	13.4	19.9	52.3	6.7	85.6
Bunny model (Figure 9c)	0.05	56k	10000	12.9	15.8	25.5	1	54.2
Bunny model (Figure 10b)	0.5	100k	10000	25.2	41.6	196.1	16.3	262.9
Bunny model (Figure 10c)	0.5	200k	10000	45.7	73.1	452.7	21.1	571.5

Table 2: Simulation performance. The table contains the time step size, the number of particles, the viscosity coefficient, the average computation times for neighborhood search, pressure solver (DFSPH) and our viscosity solver, the number of iterations required by the viscosity solver and the total computation time per simulation step. All times are given in milliseconds. In case of adaptive time stepping (CFL) the table contains the average time step size.

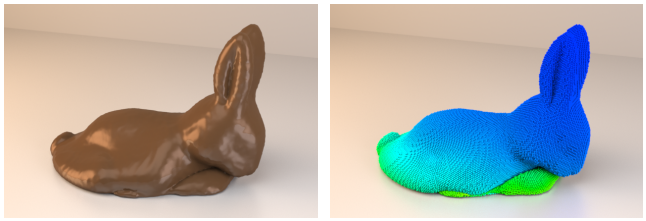


Figure 11: Melting chocolate bunny. Left: Rendered mesh. Right: Particles with color-coded temperatures.

viscosity solver requires no additional memory since it works directly on the velocity field without storing intermediate results.

Temperature-dependent viscosity Finally, we demonstrate that our method is able to simulate realistic melting effects using our extension of temperature-dependent viscosity (see Section 5). Figure 11 shows a melting chocolate bunny on a warm floor and the color-coded temperatures in the fluid particles. The accompanying video contains another simulation with a melting bunny where a point heat source is located above the bunny. At the beginning of both simulations the viscosity coefficient of all particles is set to a high value of $\mu = 5 \cdot 10^6 \frac{\text{kg}}{\text{m} \cdot \text{s}}$ to simulate an almost rigid bunny. Then due to the heat transfer the viscosity coefficient decreases so that at a temperature of 50°C the fluid has a viscosity of $\mu = 100 \frac{\text{kg}}{\text{m} \cdot \text{s}}$ which corresponds to molten chocolate. This experiment demonstrates that our method is able to handle a large range of viscosities.

8. Conclusion

In this paper a novel implicit solver for the simulation of highly viscous fluids using SPH was presented. As a result of an in-depth analysis of the most recent implicit approaches, a novel formulation was developed that fulfills all requirements to realistically simulate viscous flow. We have demonstrated that our method is able to surpass the existing approaches in physical accuracy and visual realism while it has comparable performance and no additional memory requirements. By generalizing the viscosity parameter to a spatially varying field and by incorporating a discretization of the heat equation into our solver, we were also able to simulate melting.

Naturally, our method comes with some limitations. Our discretization of the Laplacian is only valid if the fluid is incompressible. More specifically, it requires that a divergence-free velocity field is enforced. However, since most fluids are almost incompressible, this not a major limitation. Another limitation is that our heat sources do not account for shadowing effects and simply project the heat waves onto the surface of the melting body.

9. Acknowledgment

The statue model is courtesy of Concept Artist Marcel Wissel (www.marcelwissel.de). The Chocolate Bunny model was

made available by Sketchfab user Louis. The work of Marcel Weiler is supported by the 'Excellence Initiative' of the German Federal and State Governments and the Graduate School of Computational Engineering at Technische Universität Darmstadt.

References

[AIA*12] AKINCI N., IHMSEN M., AKINCI G., SOLENTHALER B., TESCHNER M.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Trans. Graph. 31*, 4 (2012), 1–8. 4

[BAV*10] BERGOU M., AUDOLY B., VOUGA E., WARDETZKY M., GRINSPUN E.: Discrete viscous threads. *ACM Trans. Graph. 29*, 4 (2010), 1. 3

[BB08] BATTY C., BRIDSON R.: Accurate viscous free surfaces for buckling, coiling, and rotating liquids. In *ACM SIGGRAPH/Eurographics SCA* (2008), pp. 219–228. 2

[Ben17] BENDER J.: SPlisHSPlasH Library. https://github.com/InteractiveComputerGraphics/SPlisHSPlasH, 2017. 7

[BGFAO17] BARREIRO H., GARCÍA-FERNÁNDEZ I., ALDUÁN I., OTADUY M. A.: Conformation constraints for efficient viscoelastic fluid simulation. *ACM Trans. Graph. 36*, 6 (2017), 221.1–221.11. 2, 3, 5, 6, 7

[BH11] BATTY C., HOUSTON B.: A Simple Finite Volume Method for Adaptive Viscous Liquids. In *ACM SIGGRAPH/Eurographics SCA* (2011), pp. 111–118. 2

[BK17] BENDER J., KOSCHIER D.: Divergence-Free SPH for Incompressible and Viscous Fluids. *IEEE Trans. on Visualization and Computer Graphics 23*, 3 (2017), 1193–1206. 2, 3, 6, 7, 8, 9

[BKKW17] BENDER J., KOSCHIER D., KUGELSTADT T., WEILER M.: A Micropolar Material Model for Turbulent SPH Fluids. In *ACM SIGGRAPH/Eurographics SCA* (2017), pp. 1–8. 3

[Bro85] BROOKSHAW L.: A method of calculating radiative heat diffusion in particle simulations. *Publications of the Astronomical Society of Australia 6*, 2 (1985), 207–210. 2, 4, 5

[BUAG12] BATTY C., URIBE A., AUDOLY B., GRINSPUN E.: Discrete Viscous Sheets. *ACM Trans. Graph. 31*, 4 (2012), 1–7. 3

[CBP05] CLAVET S., BEAUDOIN P., POULIN P.: Particle-based viscoelastic fluid simulation. In *ACM SIGGRAPH/Eurographics SCA* (2005), pp. 1–11. 3

[CM99] CLEARY P. W., MONAGHAN J. J.: Conduction modelling using smoothed particle hydrodynamics. *Journal of Computational Physics 148*, 1 (1999), 227 – 264. 2, 4

[dSP*14] DE SOUZA ANDRADE L. F., SANDIM M., PETRONETTO F., PAGLIOSA P., PAIVA A.: SPH Fluids for Viscous Jet Buckling. In *IEEE SIBGRAPI Graphics, Patterns and Images* (2014), pp. 65–72. 2

[ER03] ESPANOL P., REVENGA M.: Smoothed dissipative particle dynamics. *Physical Review E 67*, 2 (2003). 4

[FM96] FOSTER N., METAXAS D.: Realistic animation of liquids. *Graphical Models and Image Processing 58*, 5 (sep 1996), 471–483. 2

[FMH*94] FLEBBE O., MUENZEL S., HEROLD H., RIFFERT H., RUDER H.: Smoothed Particle Hydrodynamics: Physical viscosity and the simulation of accretion disks. *The Astrophys. Journal 431* (1994). 4

[HMG*06] HABIBI M., MALEKI M., GOLESTANIAN R., RIBE N. M., BONN D.: Dynamics of liquid rope coiling. *Physical Review E 74* (2006). 7

[ICS*14] IHMSEN M., CORNELIS J., SOLENTHALER B., HORVATH C., TESCHNER M.: Implicit incompressible SPH. *IEEE Trans. on Visualization and Computer Graphics 20*, 3 (2014), 426–435. 3

[IOS*14] IHMSEN M., ORTHMANN J., SOLENTHALER B., KOLB A., TESCHNER M.: SPH Fluids in Computer Graphics. *Eurographics (State of the Art Reports)* (2014), 21–42. 4, 5

[JSD04] JUBELGAS M., SPRINGEL V., DOLAG K.: Thermal conduction in cosmological SPH simulations. *Monthly Notices of the Royal Astronomical Society 351*, 2 (2004), 423–435. 4

[JWJ*14] JONES B., WARD S., JALLEPALLI A., PERENIA J., BARGTEIL A. W.: Deformation embedding for point-based elastoplastic simulation. *ACM Trans. Graph. 33*, 2 (2014), 1–9. 3

[KB17] KOSCHIER D., BENDER J.: Density Maps for Improved SPH Boundary Handling. In *ACM SIGGRAPH/Eurographics SCA* (2017), pp. 1–10. 2

[Lau11] LAUTRUP B.: *Physics of Continuous Matter*. Taylor & Francis, 2011. 3

[LBB17] LARIONOV E., BATTY C., BRIDSON R.: Variational stokes: A unified pressure-viscosity solver for accurate viscous liquids. *ACM Trans. Graph. 36*, 4 (2017). 2

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-Based Fluid Simulation for Interactive Applications. In *ACM SIGGRAPH/Eurographics SCA* (2003), pp. 154–159. 2, 3

[MM97] MORRIS J., MONAGHAN J.: A Switch to Reduce SPH Viscosity. *Journal of Computational Physics 136*, 1 (1997), 41–50. 2

[MM13] MACKLIN M., MÜLLER M.: Position Based Fluids. *ACM Trans. Graph. 32*, 4 (2013), 1–5. 2, 3

[MMC16] MACKLIN M., MÜLLER M., CHENTANEZ N.: XPBD: Position-based Simulation of Compliant Constrained Dynamics. In *ACM Motion in Games* (2016), ACM, pp. 49–54. 3, 6

[Mon92] MONAGHAN J.: Smoothed Particle Hydrodynamics. *Annual Review of Astronomy and Astrophysics 30*, 1 (1992), 543–574. 2, 4, 6

[Mon05] MONAGHAN J. J.: Smoothed Particle Hydrodynamics. *Reports on Progress in Physics 68*, 8 (2005), 1703–1759. 2, 4

[PICT15] PEER A., IHMSEN M., CORNELIS J., TESCHNER M.: An Implicit Viscosity Formulation for SPH Fluids. *ACM Trans. Graph. 34*, 4 (2015), 1–10. 2, 3, 4, 5, 6, 7, 8, 9

[PPLT09] PAIVA A., PETRONETTO F., LEWINER T., TAVARES G.: Particle-based viscoplastic fluid/solid simulation. *Computer-Aided Design 41*, 4 (2009), 306–314. 2

[Pri12] PRICE D. J.: Smoothed particle hydrodynamics and magnetohydrodynamics. *J. Comput. Phys. 231*, 3 (2012), 759–794. 4

[PT16] PEER A., TESCHNER M.: Prescribed velocity gradients for highly viscous SPH fluids with vorticity diffusion. *IEEE Trans. on Visualization and Computer Graphics* (2016), 1–9. 2, 3, 4, 5, 6, 7, 8, 9

[REN*04] RASMUSSEN N., ENRIGHT D., NGUYEN D., MARINO S., SUMNER N., GEIGER W., HOON S., FEDKIW R.: Directable photorealistic liquids. In *ACM SIGGRAPH/Eurographics SCA* (2004). 2

[SB12] SCHECHTER H., BRIDSON R.: Ghost SPH for animating water. *ACM Trans. Graph. 31*, 4 (2012), 61:1–61:8. 2

[Sta99] STAM J.: Stable Fluids. In *ACM Conference on Computer Graphics and Interactive Techniques* (1999), pp. 121–128. 2

[TDF*15] TAKAHASHI T., DOBASHI Y., FUJISHIRO I., NISHITA T., LIN M.: Implicit Formulation for SPH-based Viscous Fluids. *Computer Graphics Forum 34*, 2 (2015), 493–502. 2, 3, 4, 5, 6, 7, 8, 9

[TDFN14] TAKAHASHI T., DOBASHI Y., FUJISHIRO I., NISHITA T.: Volume preserving viscoelastic fluids with large deformations using position-based velocity corrections. *The Visual Computer* (2014). 3

[TNF14] TAKAHASHI T., NISHITA T., FUJISHIRO I.: Fast simulation of viscous fluids with elasticity and thermal conductivity using position-based dynamics. *Computers & Graphics 43*, 1 (2014), 21–30. 3

[WBF*96] WATKINS S. J., BHATTAL A. S., FRANCIS N., TURNER J. A., WHITWORTH A. P.: A new prescription for viscosity in smoothed particle hydrodynamics. *Astron. Astrophys. Suppl. Ser. 119*, 1 (1996). 4

[ZLQF15] ZHU B., LEE M., QUIGLEY E., FEDKIW R.: Codimensional Non-Newtonian Fluids. *ACM Trans. Graph. 34*, 4 (2015), 1–9. 3