

Adapted SIMPLE Algorithm for Incompressible SPH Fluids With a Broad Range Viscosity

Shusen Liu¹, Xiaowei He¹, Wencheng Wang¹, *Member, IEEE*, and Enhua Wu², *Member, IEEE*

Abstract—In simulating viscous incompressible SPH fluids, incompressibility and viscosity are typically solved in two separate stages. However, the interference between pressure and shear forces could cause the missing of behaviors that include preservation of sharp surface details and remarkable viscous behaviors such as buckling and rope coiling. To alleviate this problem, we introduce for the first time the semi-implicit method for pressure linked equations (SIMPLE) into SPH to solve incompressible fluids with a broad range viscosity. We propose to link incompressibility and viscosity solvers, and impose incompressibility and viscosity constraints iteratively to gradually remove the interference between pressure and shear forces. We will also discuss how to solve the particle deficiency problem for both incompressibility and viscosity solvers. Our method is stable at simulating incompressible fluids whose viscosity can range from zero to an extremely high value. Compared to state-of-the-art methods, our method not only produces realistic viscous behaviors, but is also better at preserving sharp surface details.

Index Terms—Smoothed particle hydrodynamics (SPH), SIMPLE algorithm, fluid simulation, incompressibility, viscosity

1 INTRODUCTION

VISCOUS fluids can be found everywhere in our daily life, such as honey, oil and ketchup, etc. Although smoothed particle hydrodynamics (SPH) has been widely used to model low viscous or nearly inviscid incompressible fluids [1], [2], how to correctly model highly or even dynamic viscous incompressible fluids still faces challenges. Since explicit methods can hardly be extended for modeling highly viscous fluids due to the time step restriction [3], [4], recent works prefer to solve fluid viscosity implicitly [5], [6], [7]. According to the analysis in [7], an ideal implicit viscosity solver should satisfy several requirements as listed in their work. However, in simulating highly viscous incompressible fluids, an additional requirement, i.e., incompressibility and viscosity constraints should not interfere [6], [8], is also necessary to be fulfilled.

In traditional projection-based particle fluid solvers, the most common way to solve the governing equations of

viscous incompressible fluids is to apply operator splitting, where the influence of pressure and shear forces are considered independently [9], [10]. While operator splitting has the advantage of reducing computational cost and simplifying code implementation, Larionov *et al.* [8] pointed out splitting viscosity from projection will cause the loss of behaviors that depend on coupling between pressure and shear forces, such as preservation of sharp surface details and the rope coiling effect. This is because when both incompressibility and viscosity constraints are solved independently but only once, at least one of the two constraints will be damaged by the other at the end of each simulation step. Therefore, referring to fluid-solid coupling method in [11], a natural question is why not select an existing incompressibility solver and an existing viscosity solver, then iterate over them multiple times to alleviate the interference artifact? Unfortunately, according to our test in Figs. 7, 8 and 13c, where the DFSPH incompressibility solver [12] and the implicit viscosity solver [7] are selected, this strategy does not work well. The reason could be that if incompressibility and viscosity constraints are solved independently, the interference between pressure and shear forces could introduce drift errors to each other. Peer *et al.* [6], [13] proposed an alternative way to solve the problem caused by the interference. However, the constructed velocity field suffers from non-physical velocity damping [7]. Our tests show that their method fails to reproduce the correct coiling effect, as shown in Fig. 8.

Motivated by the *semi-implicit method for pressure linked equations* (SIMPLE) [14], we propose a novel iterative strategy for simulating viscous incompressible SPH fluids, with which the interference artifact can be addressed. Unlike the above mentioned simple strategy, the incompressibility and viscosity solvers in our method are linked. In solving the momentum equation for viscosity, aside from the velocity field, a term involving the pressure gradient is also

- Shusen Liu is with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100864, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: liuss@ios.ac.cn.
- Xiaowei He is with the Beijing Key Lab of Human-Computer Interaction, Institute of Software, Chinese Academy of Sciences, Beijing 100864, China. E-mail: xiaowei@iscas.ac.cn.
- Wencheng Wang is with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100864, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: whn@ios.ac.cn.
- Enhua Wu is with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100864, China, and also with the University of Macau, Macau 999078, China. E-mail: ehwu@umac.mo.

Manuscript received 29 Apr. 2020; revised 24 Jan. 2021; accepted 25 Jan. 2021. Date of publication 1 Feb. 2021; date of current version 1 Aug. 2022.
(Corresponding authors: Xiaowei He and Wencheng Wang.)
Recommended for acceptance by M. Teschner.
Digital Object Identifier no. 10.1109/TVCG.2021.3055789

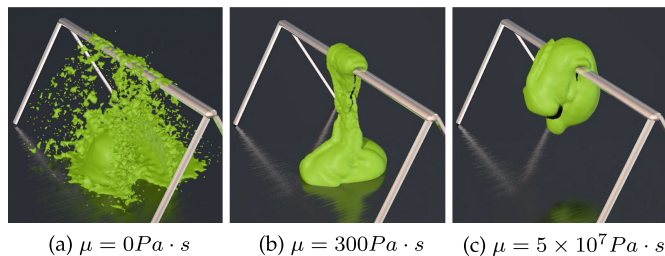


Fig. 1. Newtonian fluids. Three bunnies with different viscosities are dropped onto a bar. This example shows that our method is stable at simulating incompressible fluids with a viscosity that can range from zero to an extremely high value.

integrated. This guarantees that the effect of pressure is properly considered when fluid viscosity is solved. In solving the momentum equation for incompressibility, the linked viscosity term will be discarded due to the convergence characteristics of SIMPLE algorithm. Therefore, a traditional projection-based incompressibility solver still can be used to solve the pressure Poisson equation. When incompressibility and viscosity solvers are taken iteratively, our method is able to converge to a globally optimal solution. Therefore, remarkable viscous behaviors, such as coiling and buckling, can be modeled with a good property of preserving sharp surface details. Besides, our method is stable at simulating incompressible Newtonian and non-Newtonian fluids with any given viscosity, as demonstrated in Figs. 1 and 2.

2 RELATED WORKS

Viscosity Models. Many forms of different viscosity models have been applied in computer graphics. We only review the most commonly used ones here. The first category uses the finite different method to discretize the viscosity term [15], which will be referred to as the *local viscosity model* considering its similarity to the Laplacian operator proposed by [16]. An explicit Euler integration method was used to impose viscosity in [4], [17], [18]. To enable stable simulation of highly viscous fluids, Weiler *et al.* [7] for the first time proposed an implicit method to solve viscosity. Compared to other viscosity models, the local model have a number of desirable features including being Galilean invariant, linear and angular momentum conservation. Besides, it does not suffer from the particle deficiency problem. The second category is labelled as the *global viscosity model*, which typically requires to take into account second-ring neighbors to calculate viscous forces. Since viscosity and incompressibility constraints can interfere, Peer *et al.* [6] proposed to project the velocity gradient to a target that does not only encode the desired viscosity, but also preserves arbitrary velocity divergences. Takahashi *et al.* [5] solved the exact viscosity implicitly to enable the usage of larger time steps and higher viscosities. Bender and Koschier [12] also proposed an implicit viscosity solver by using a parallel Jacobi method to enforce a target strain rate. Peer and Teschner [13] extended their earlier work to preserve the vorticity for highly viscous fluids. One disadvantage of the exact viscosity model is that artificial viscosity will be introduced to boundary due to the particle deficiency problem [7]. The last category is non-physical viscosity model, which is labelled as the *XSPH artificial viscosity model*.

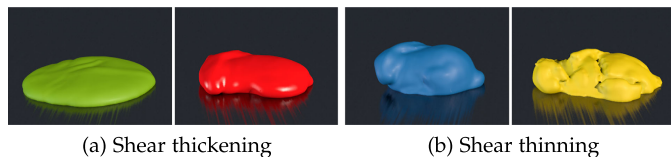


Fig. 2. Non-Newtonian fluids. (a) A low viscosity ($\mu = 50 \text{ Pa} \cdot \text{s}$) Newtonian bunny (left) and a shear thickening bunny (right) with equal initial viscosities are dropped onto the ground. The motion of the shear thickening bunny slows down upon impact, but then restores as its shear rate decreases; (b) A high viscosity ($\mu = 5 \times 10^5 \text{ Pa} \cdot \text{s}$) Newtonian bunny (left) and a shear thinning bunny (right) with equal initial viscosities are dropped onto the ground. The motion of the shear thinning bunny accelerates upon impact.

It was originally proposed by Monaghan [19] to prevent penetration for particles impinging on each other. Schechter and Bridson [20] introduced this technique into computer graphics to stabilize the ghost SPH method. Macklin and Müller [21] applied XSPH viscosity in a position-based fluid solver. Although Müller *et al.* [22] proposed to use a slightly different formulation to model viscosity, their formulation still can be regarded as an XSPH artificial viscosity model as both models have the same physical meanings. Other works applying this method include [23], [24], [25]. Its wide usage attributes to the good performance in stability. However, the XSPH artificial viscosity model fails to reproduce certain physical features. For example, it cannot conserve angular momenta and introduces artificial viscosity even for rigid body rotation.

Incompressibility Model. We categorize all incompressibility solvers in SPH into two main classes. One aims to solve fluid incompressibility that targets a desired density. In the early work, incompressible fluids are assumed to be slightly compressible. Therefore, a equation of state (EOS) was applied to model the relationship between pressure and density [4], [26], [27]. However, since EOS-based solvers are subject to the limitation of small time steps, Solenthaler and Pajarola [1] first proposed an iterative predictive-corrective scheme to enforce a constant density. Continuous efforts on accelerating the convergence rate were taken by several later works [2], [28]. The other class aims to solve fluid incompressibility that targets the divergence-free condition. Cummins and Rudman [16] first proposed an approximate projection method to enforce incompressibility. Motivated by the staggered-grid method, He *et al.* [10] proposed a staggered particle method to simulate fluid-solid coupling. Bender and Koschier [12] proposed to fulfil the divergence-free condition in a way similar to the iterative EOS-based solvers. Among all solvers, the velocity-constraint incompressibility model are preferable for simulating viscous fluids because undesired bulk viscosity could arise if the divergence-free condition is not fulfilled [6], [29]. For more discussions on incompressibility solvers, we refer to the review papers [30], [31], [32].

The Linkage Between Solvers. In simulating highly viscous incompressible fluids with large time steps, the interference between pressure and viscosity solvers cannot be ignored [6]. Under the Eulerian frame, this problem has been studied more thoroughly [8], [14]. However, only a few works have been done to address this issue under the Lagrangian frame. Peer *et al.* [6], [13] obtained the viscosity constraint by projecting the velocity gradient to a target and reconstructing the velocity field. This approach is able to avoid the interference,

yet a significant damping is introduced. Additionally, their approach cannot simulate low viscous fluids realistically [7]. Aside from viscous incompressible fluids, the interference problem also exists in solid-fluid coupling or air-liquid coupling. Gissler *et al.* [11] proposed to resolve the instability in fluid-rigid coupling or rigid-rigid coupling with SPH. Batty *et al.* [33] reformulated solid-fluid coupling as a kinetic energy minimization problem, achieving a stable and accurate unified two-way coupling solver. Aanjaneya *et al.* [34] adopted this similar idea, but proposed a monolithic solver to simulate bubbles. To realize Euler-Lagrange solid-fluid coupling, Robinson-Mosher *et al.* [35] proposed a novel method that handles solid-fluid coupling in a fully implicit manner to obtain better stability. Robinson-Mosher *et al.* [36] later extended their work to obtain a symmetric formulation of the coupled system, making it easier to solve.

3 OVERVIEW

The governing equations for a viscous incompressible fluid are expressed in the Lagrangian form as

$$\rho \frac{D\mathbf{v}}{Dt} = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \mathbf{f}, \quad (1)$$

$$\nabla \cdot \mathbf{v} = 0, \quad (2)$$

where \mathbf{v} is velocity, ρ is density, p is pressure, $\boldsymbol{\tau} = \mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)$ is deviatoric stress tensor, μ is the dynamic viscosity coefficient, \mathbf{f} is external force per unit volume. We assume ρ to be constant and μ be varying in space. To better understand this interference issue between incompressibility and viscosity in operator splitting schemes, we take the prediction-correction scheme applied in [9] for a demonstration. In the prediction step, an explicit integration without enforcing incompressibility is taken, i.e., only shear and body forces are used to update the velocity field. Equation (1) should be fulfilled at the end of the prediction step, while the divergence-free condition is not fulfilled yet. Then, a pressure Poisson equation is solved to enforce the divergence-free condition in Equation (2). Unfortunately, after the velocity field is updated with the pressure force, Equation (1) does not stand any longer due to the changes in the shear force and velocity.

Our purpose is to guarantee the final status of p and \mathbf{v} at the end of each simulation step fulfill both governing equations. Despite the dependence of μ on space, we make an assumption that $\nabla \cdot (\mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)) = \mu\nabla^2\mathbf{v} + \mu\nabla \cdot (\nabla\mathbf{v}^T)$. The second term just cancels out for incompressible fluids due to the relationship $\mu\nabla \cdot (\nabla\mathbf{v}^T) = \mu\nabla(\nabla \cdot \mathbf{v}) = 0$. After taking an implicit backward Euler method in time, the momentum and continuity equations are simplified as

$$\rho \frac{\mathbf{v}^{n+1} - \mathbf{v}^*}{\delta t} = -\nabla p^{n+1} + \mu\nabla^2\mathbf{v}^{n+1}, \quad (3)$$

$$\nabla^2 p^{n+1} = \frac{\rho}{\delta t} \nabla \cdot \mathbf{v}^* + \nabla \cdot (\mu\nabla^2\mathbf{v}^{n+1}), \quad (4)$$

where \mathbf{v}^* is the intermediate velocity after adding external forces, δt is time step size, superscript $n+1$ indicates the new value at the end of each simulation step. The difficulty in solving Equations (3) and (4) lies in that the unknown pressure

p^{n+1} and velocity \mathbf{v}^{n+1} are coupled for both equations. Even worse, the value of $\nabla \cdot (\mu\nabla^2\mathbf{v}^{n+1})$ could be quite sensitive to particle distributions due to the third order derivative of \mathbf{v} .

To solve the above-mentioned issue, a well-known SIMPLE algorithm was originally proposed in the finite volume method (FVM) [14]. Before we go into the details on how to extend the SIMPLE algorithm to SPH, let us review the fundamental principles of the SIMPLE method. To solve Equations (3) and (4), the SIMPLE algorithm starts with a guessed pressure field p^* . The momentum equation for viscosity can then be solved as follows:

$$\rho \frac{\mathbf{v}' - \mathbf{v}^*}{\delta t} = -\nabla p^* + \mu\nabla^2\mathbf{v}'. \quad (5)$$

However, unless the correct pressure is given, the resulting velocity field \mathbf{v}' will not satisfy the divergence-free condition in Equation (2), i.e., $\nabla \cdot \mathbf{v}' \neq 0$. Therefore, we should find a way to improve p^* , so that the velocity field can get closer to satisfying the divergence-free condition. Let us denote the pressure correction as p' , the new velocity \mathbf{v}'' after correction is expressed as

$$\mathbf{v}'' = \mathbf{v}' - \frac{\delta t}{\rho} \nabla p', \quad (6)$$

where \mathbf{v}'' satisfies $\nabla \cdot \mathbf{v}'' = 0$. By taking the divergence of both sides, we have

$$\nabla \cdot \frac{\delta t}{\rho} \nabla p' = \nabla \cdot \mathbf{v}', \quad (7)$$

which is the same as [29].

Compared to Equation (4), we have dropped the viscosity term from the above equation. As suggested by Patankar [14], the omission of the viscosity term is absolutely unacceptable if the ultimate solution would not be the true solution of the discretized momentum and continuity equations. Fortunately, as we solve Equations (5) and (7) iteratively, both the velocity and pressure fields can progressively converge to their final solutions (refer to Fig. 13a). That is to say, the global solution is actually uninfluenced by the approximation of dropping the viscosity term. Besides, above simplification is essential to extend the SIMPLE algorithm to SPH because the viscosity term will otherwise introduce a third-order differential operator, which is difficult to discretize, to SPH. Please note that the word ‘semi-implicit’ in the name SIMPLE is used to acknowledge the omission of the viscosity term in Equation (6). For completeness, we list the full operations of SIMPLE algorithm as follows:

- 1) Guess a pressure field p^* ;
- 2) Solve the momentum Equation (5) to obtain \mathbf{v}' ;
- 3) Solve the pressure Poisson Equation (7) to obtain p' ;
- 4) Add p' to p^* ;
- 5) Return to step 2 and repeat the whole procedure until the converged solution is obtained.

Note the rate of the convergence depends on the particular formulation of the p' equation [14]. If too many terms are omitted, divergence may result. For example, if the fluid incompressibility is solved with the DFSPH method, divergence occurs due to the omission of the pressure field, as shown in Fig. 13c.

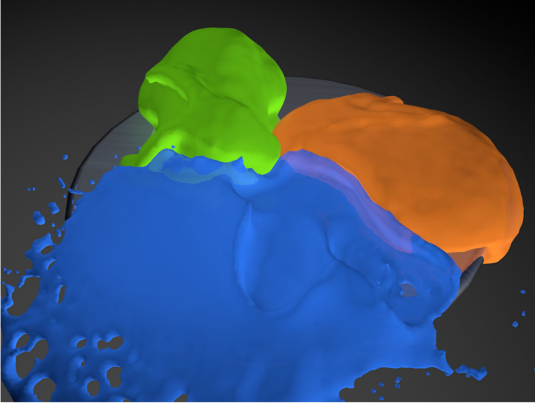


Fig. 3. Three bunnies with different viscosities are dropped onto a table to demonstrate their interactions.

In the following section, we will discuss how to apply the SIMPLE algorithm to solve viscous incompressible fluids on particles.

4 EXTENDING THE SIMPLE ALGORITHM TO SPH

Since the SIMPLE algorithm was originally proposed under the Eulerian framework, whether it works or helps to improve the accuracy for particle-based fluid simulation remains uninvestigated yet. The difficulty lies in how to address the sensitive particle distribution and particle deficiency problems for particle-based fluid simulation.

Considering a fluid that is discretized into a set of particles, each particle is associated with a lumped mass m_i , volume V_i and position \mathbf{x}_i . The particle approximation of an arbitrary function ϕ can be written in a summation of the neighboring particles as follows:

$$\phi_i = \sum_j V_j \phi_j W_{ij}, \quad (8)$$

where W_{ij} denotes the smoothing function $W(\|\mathbf{x}_i - \mathbf{x}_j\|, h)$, h is the smoothing length. To derive SPH formulations for gradient and Laplacian operators, there in fact exist a number of different ways [37], [38]. To discretize the pressure gradient, the symmetric repulsive pressure gradient model is mostly used because of its momentum conservation feature. However, this model is sensitive to the tensile instability problem [31]. In order to capture realistic stretching behaviors in viscous fluids (e.g., Fig. 4), we choose to discretize the gradient operator with the Taylor-series consistent model

$$\nabla_i \phi = - \sum_j V_j \phi_{ij} \nabla_i W_{ij}, \quad (9)$$

where ϕ_{ij} is short for $\phi_i - \phi_j$, $\nabla_i W_{ij} = \frac{\mathbf{x}_i - \mathbf{x}_j}{r_{ij}} \frac{\partial W_{ij}}{\partial r_{ij}}$ and $r_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$. Compared to the symmetric repulsive model, the asymmetric one is more accurate as it recovers a zeroth order accuracy. Besides, it is better at preserving features that undergo stretching [39]. For the Laplacian operator, its standard form based on a finite difference discretization is written as

$$\nabla_i^2 \phi = \sum_j V_j \frac{\phi_{ij} \cdot \mathbf{r}_{ij}}{r_{ij}^2 + \eta^2} \nabla_i W_{ij}, \quad (10)$$

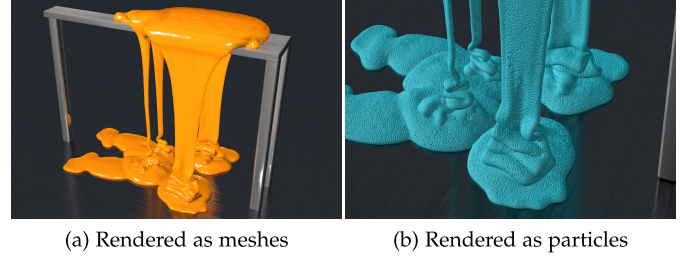


Fig. 4. An armadillo consisting of 462k particles is dropped onto a bench. Our method is able to reproduce remarkable viscous behaviors, such as rope coiling and buckling, as well as other small-scale thin features, such as filaments and sheets.

where η is a small value to prevent singularities and $\mathbf{r}_{ij} = \mathbf{x}_i - \mathbf{x}_j$.

4.1 Viscosity

To solve the momentum Equation (5), both ∇p^* and $\nabla^2 \mathbf{v}'$ should be discretized first. If a particle's location is far from free surface boundary, i.e., its support domain is not truncated by the boundary, we can easily get the discretized forms by inserting p^* and \mathbf{v}' into Equations (9) and (10). Unfortunately, if the particle is near the free surface boundary, both terms suffer from the particle deficiency problem. To investigate this problem, we illustrate a one-dimensional linear function in Fig. 5a. The corresponding analytic formula and numerical calculation of its first and second order derivatives are demonstrated in Fig. 5b. We can notice the particle approximation no longer reproduces the correct first and second order derivatives for boundary particles.

To resolve the particle deficiency problem, one could introduce ghost air particles, as was done in [20]. However, creating ghost air particles requires more memory and computational resources.

Fortunately, a special treatment of free surface boundary conditions in fluids allows us to solve the particle deficiency problem efficiently with a semi-analytical approach [39], [41], [42]. Following their work, if liquid particles j^l and ghost air particles j^a are uniformly distributed, we can get the following relationship:

$$\begin{aligned} \sum_j V_j \nabla_i W_{ij} &= \sum_{j^l} V_{j^l} \nabla_i W_{ij^l} + \sum_{j^a} V_{j^a} \nabla_i W_{ij^a} \\ &= \nabla 1 = 0. \end{aligned} \quad (11)$$

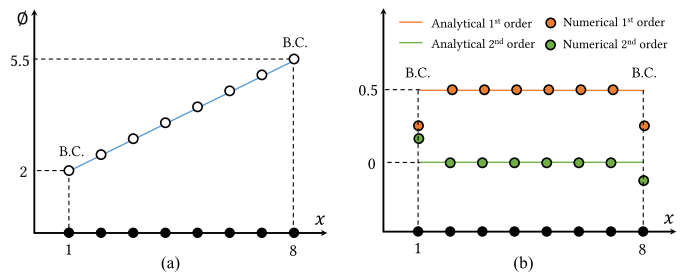


Fig. 5. A linear function used to demonstrate the particle deficiency problem in SPH. We select the Cubic spline function with $h = 2$ as the smoothing length. (a) A linear function and the corresponding particle representation; (b) Analytical formula and numerical calculations of the first and second derivatives of the linear function.

Therefore, the pressure gradient can be discretized as

$$\nabla_i p^* = \sum_{j^l} V_{j^l} p_{j^l}^* \nabla_i W_{ij^l} + \sum_{j^a} V_{j^a} p_{j^a}^* \nabla_i W_{ij^a} - p_i^* \sum_j V_j \nabla_i W_{ij}, \quad (12)$$

where we have assumed all virtual ghost air particles j^a have a constant atmosphere pressure p_{atm} (which is typically set to $p_{ja}^* = p_{atm} = 0$). The velocity of virtual ghost air particles \mathbf{v}_{ja} is assumed to be equal to its central fluid particle i , i.e., $\mathbf{v}_{ja} = \mathbf{v}_i$. This has the advantage that no artificial damping will be introduced into fluids by virtual ghost air particles. By substituting (11) into (12) and setting p_{ja}^* to zero, the pressure gradient can be derived as [39]

$$\nabla_i p^* = \begin{cases} \sum_{j^l} V_{j^l} (p_{j^l}^* - p_i^*) \nabla_i W_{ij^l}, & i \notin \mathcal{B} \\ \sum_{j^l} V_{j^l} p_{j^l}^* \nabla_i W_{ij^l}, & i \in \mathcal{B} \end{cases}, \quad (13)$$

where \mathcal{B} represents the boundary particles whose support domain intersect with the boundary as Fig. 10. How to identify interior particles will be left to the next section.

Similarly, we discretize the Laplacian of the velocity field as

$$\nabla_i^2 \mathbf{v}' = 2(d+2) \sum_{j^l} V_{j^l} \frac{\mathbf{v}'_{j^l} \cdot \mathbf{r}_{ij^l}}{|\mathbf{r}_{ij^l}|^2 + \eta^2} \nabla W_{ij^l}. \quad (14)$$

The derivation of Equation (14) can be found in [3] or [32]. Note that the contributions from virtual ghost air particles for both the pressure gradient and Laplacian of velocity have been semi-analytically eliminated. In fact, the formulation in Equation (14) is exactly the same as the one used in [7]. Yet our derivation has provided a better motivation for its effectiveness at reducing visual artifacts caused by the particle deficiency problem.

Based on the above discretization scheme, the momentum Equation (5) can be divided into the following two steps

$$\mathbf{v}^{**} = \mathbf{v}^* - \frac{\delta t}{\rho} \nabla p^*, \quad (15)$$

$$\mathbf{v}' = \mathbf{v}^{**} + \frac{\delta t \mu}{\rho} \nabla^2 \mathbf{v}'. \quad (16)$$

Note Equation (16) is the same as the one solved in [7]. The calculated solution is then substituted into the pressure Poisson equation to enforce fluid incompressibility.

4.2 Incompressibility

To guarantee the convergence, the incompressibility solver should be properly derived from the pressure Poisson equation, and be able to impose the divergence-free condition on the velocity field (as discussed in Section 3). We therefore prefer to apply an approximate projection method to solve the fluid incompressibility.

Solving the pressure Poisson Equation (7) involves discretizing both the gradient and Laplacian operators as well. Besides, free-surface boundary conditions, which affect the convergence in solving the pressure Poisson equation [43],

should also be considered. In a projection based method, the pressure Poisson Equation (7) is to be transformed into a linear system in the form of $Ax = b$. The coefficient matrix A is symmetric, but cannot be guaranteed to be positive definite if no free boundary condition is imposed. For example, if a particle i has no neighbors, the corresponding diagonal entry of A is zero, indicating the coefficient matrix is singular. To resolve this problem, we introduce ghost air particles at the free surface boundary [42]. With enough ghost air particles, a particle at the free surface boundary should have full neighbors. By imposing the Dirichlet boundary condition, i.e., $p_{ja}^* = 0$ for ghost air particles, the coefficient matrix A is guaranteed to be positive definite. Therefore, typical iterative methods such as the conjugate gradient method can applied to solve the linear system. In the following context, we will give more details on how to discretize Equation (7) and impose the free-surface boundary condition.

Integrating virtual ghost air particles, the velocity divergence can be easily derived as

$$\nabla_i \cdot \mathbf{v}' = \sum_{j^l} V_{j^l} (\mathbf{v}'_{j^l} - \mathbf{v}'_i) \cdot \nabla_i W_{ij^l}. \quad (17)$$

However, discretizing the Laplacian of pressure will not be an easy task because we do not know how to calculate $\nabla_i W_{ij^a}$ if ghost air particles are not created. Following [41], we apply the semi-analytical method to discretize the Laplacian of pressure. The neighbors of particle i could contain both particles j^l and ghost air particles j^a . Since the pressure value of ghost air particles p_{ja}^* can be set to zero according to Dirichlet boundary conditions [44], we can discretize the Laplacian of pressure as

$$\nabla_i^2 p' = - \sum_{j^l} a_{ij^l} p'_{j^l} - \sum_{j^a} a_{ija} p'_{ja} + \sum_{j^l} a_{ij^l} p'_{j^l}, \quad (18)$$

where $a_{ij} = -\frac{V_j}{r_{ij} + \eta^2} \frac{\partial W_{ij}}{\partial r_{ij}}$ is the coefficient. After rearrangement for different cases, the discretized Laplacian of pressure is written as

$$\nabla_i^2 p' = \begin{cases} \sum_{j^l} a_{ij^l} (p'_{j^l} - p'_i), & i \notin \mathcal{B} \\ -A_0 p'_i + \sum_{j^l} a_{ij^l} p'_{j^l}, & i \in \mathcal{B} \end{cases}, \quad (19)$$

where A_0 is a predefined value of $A_i = \sum_{j^l} a_{ij^l}$ calculated for a particle with full neighbors at the beginning of simulation (e.g., by choosing an interior particle whose support domain is not clumped by boundary), as shown in Fig. 10a. We identify a fluid particle i as a boundary particle only if $A_i < A_0$ (Fig. 10b). Therefore, the boundary particle set \mathcal{B} is defined as

$$\mathcal{B} = \{i \mid A_i < A_0\}. \quad (20)$$

Another issue in solving fluid incompressibility with Equation (7) is the particle density drifting problem [46], because the velocity divergence-free condition cannot correct cumulative density errors. Motivated by the error compensating source (ECS) used in [47] and [48], we solve this problem by adding the following term

$$\Lambda_i = \alpha \frac{\rho_i - \rho_0}{\rho_0 \delta t}, \quad (21)$$

to the right-hand side of Equation (7) for particles with $\rho_i > \rho_0$, where ρ_0 represents the reference density and α is a positive constant. Equation (21) is similar to the source term of the projection method which corrects the density error [9]. When compression occurs, this term can generate repulsive forces to correct cumulative density errors. α should ensure that the value of this term is close to the right-hand side of Equation (7), otherwise, it may be ineffective or cause instability. Therefore, we typically set $\alpha = 10^6$ in our current implementation.

In calculating the pressure force, although it is common to clump particle pressures to zero [2], we retain the negative pressure to capture the fingering behavior in viscous fluids. More details on how to resolve the tensile instability problem can be found in [31], [48].

To evaluate the stability of our incompressibility solver, a dambreak case is simulated with four different time steps. As shown in Fig. 11, consistent behaviors can be noticed for all simulations.

Algorithm 1. Adapted SIMPLE Algorithm for Viscous Incompressible SPH Fluids

```

1: while  $t < t_{stop}$  do
2:    $\delta t \leftarrow CFL(\mathbf{v}_i^n)$ ;
3:   for all fluid particle  $i$  do
4:      $\mathbf{v}_i^* \leftarrow \mathbf{v}_i^n + \delta t \mathbf{F}_i^{ext}$ ;
5:      $\mathbf{x}_i^* \leftarrow \mathbf{x}_i^n + \delta t \mathbf{v}_i^*$ ;
6:   for all fluid particle  $i$  do
7:      $\mathcal{N}_i \leftarrow$  Find neighbors using  $\mathbf{x}_i^*$ ;
8:   Detect boundary particles;
9:   Set global iteration number  $iter = 0$ ;
10:  while  $iter < iter_{max}$  do
11:    for all fluid particle  $i$  do
12:      Compute velocity divergence  $\nabla_i \cdot \mathbf{v}'$ ;
13:      Add  $\Lambda_i$  to  $\nabla_i \cdot \mathbf{v}'$ ;
14:    while  $\epsilon^p > \epsilon_0^p$  do
15:      Take one CG iteration to update  $p'$ ;
16:      Compute the relative error  $\epsilon^p$  for Equation (7);
17:    for all fluid particle  $i$  do
18:      Compute  $p^* \leftarrow p'$ ;
19:    for all fluid particle  $i$  do
20:      Compute pressure gradient  $\nabla_i p^*$ ;
21:    while  $\epsilon^v > \epsilon_0^v$  do
22:      Take one CG iteration for viscosity;
23:      Compute the relative error  $\epsilon^v$  for Equation (5);
24:    for all fluid particle  $i$  do
25:      Set new velocity  $\mathbf{v}_i^{n+1} = \mathbf{v}'_i$ ;

```

4.3 Iterative Solver

To demonstrate the interference problem between incompressibility and viscosity solvers, we simulated the rope effect with two different settings. In Fig. 6a, the viscosity solver is scheduled to run first. We can notice the incompressibility solver causes significant loss of viscous behaviors, such as the rope coiling. In contrast, when the incompressibility solver is taken first, the volume of fluid cannot be well preserved, as demonstrated at bottom of Fig. 6b. To solve the interference problem, we propose to improve the original SIMPLE algorithm by solving incompressibility first. In case the divergence-free condition is

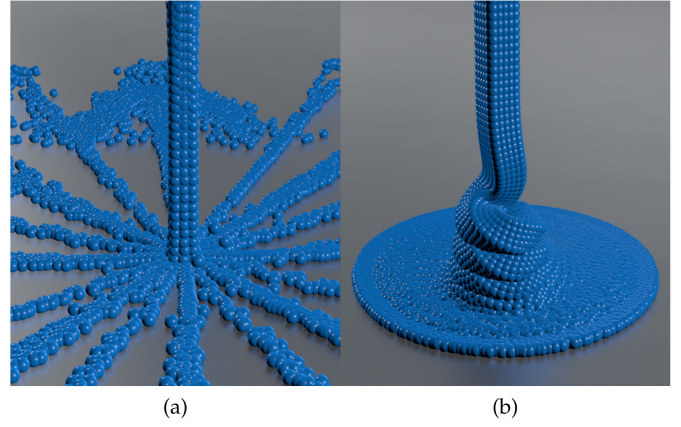


Fig. 6. A fluid with a viscosity of $\mu = 500 Pa \cdot s$ is simulated to demonstrate the interference between pressure and shear forces. (a) Remarkable viscous behaviors, such as rope coiling, are missing when incompressibility is solved after viscosity; (b) Severe volume loss is noticeable at the bottom when viscosity is solved after incompressibility.

violated by the viscosity solver, we take more overall iterations to alleviate the interference problem.

The detailed algorithm is demonstrated in Algorithm 1. It contains two loops. The two inner loops are used to solve Equations (5) and (19) independently with an iterative solver. Figs. 9a and 9b demonstrate the convergence rates of two commonly used iterative solvers, including both the Jacobi method and the conjugate gradient method. We can notice faster convergence rate of the conjugate gradient compared to the Jacobi method in solving both fluid incompressibility and viscosity. The outer loop is also taken iteratively, with a purpose to diminish the interference between pressure and viscosity.

Fig. 12 demonstrates an intuitive comparison on the effect of increasing the global iteration number, where the iteration number is increased from 1 to 16. As the global iteration number increases, we can notice a better property of shape preserving without losing viscous behaviors.

To demonstrate whether our method converges to a global solution, Fig. 13a plots the convergence curves of relative errors for both the incompressibility and viscosity solvers. Note the pressure and velocity fields are largely affected between each other at the beginning of the global iteration. However, as we take more global iterations, relative errors for both fields are gradually removed. In other words, the interference between pressure and viscosity can be alleviated with our method. Further experiments show that the global convergence rate mostly depends on the viscosity coefficient, as demonstrated in Fig. 13b. Both the time step size and the particle sampling distance only have a minor effect on the global convergence rate, as illustrated in Fig. 14. Unfortunately, if we use DFSPH [12] to solve fluid incompressibility, the algorithm fails to converge to a global solution, as shown in Fig. 13c. That is to say, the idea of coupling an arbitrary incompressibility and a viscosity solver may not be a serious solution to simulate viscous incompressible fluids.

To further evaluate the dependence of viscous behaviors on the SIMPLE iteration number, Fig. 15 demonstrates four materials with different viscosities. As the iteration number is increased, it can be noted that patterns of all four

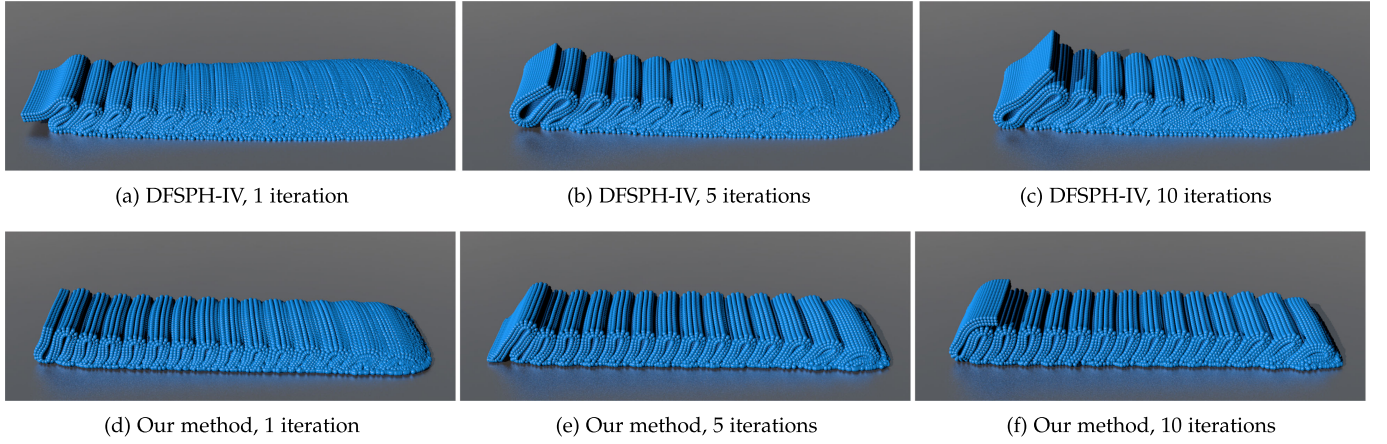


Fig. 7. Comparison of the buckling effect between our approach and state-of-the-art method [7] with different number of global iterations. DFSPH-IV means we use DFSPH [12] to solve incompressibility and the implicit method [7] to solve viscosity. For DFSPH-IV, the numbers and depths of the “wrinkles” of buckling effects change with increasing iterations, which shows that its physical viscosity changes drastically with global iterations [40], due to the divergence.

materials tend to be stable. In other words, if a sufficient iteration number is taken, real physical behaviors for all viscous fluids should be reached. However, in the practical implementation, we suggest to only take a small fixed number of iterations for efficiency. According to the comparison, 10 iterations will be enough to achieve desirable visual effects for most cases. However, for fluids with a high viscosity, e.g., $\mu > 10^4 \text{ Pa} \cdot \text{s}$, more than 10 iterations can better preserve the shapes.

5 RESULTS AND DISCUSSIONS

In this section, we simulate various fluids, including both Newtonian and non-Newtonian ones. All implementations were performed on an Intel CPU (I7-8700K 3.70GHz) with time-consuming parts (neighbor-list searching, incompressibility solver, viscosity solver and velocity updating, etc.) parallelized on an NVIDIA GPU (Geforce RTX2080). Surface

tension is based on the method of He *et al.* [42]. Surface meshes are reconstructed with the method of Bhattacharya *et al.* [49]. To simplify solid-fluid interactions, signed distance field is applied to impose a no-slip solid wall boundary condition, i.e., no relative displacements occur between the signed distance field and the particles on its surface. Time step size is fixed or restricted by the CFL condition [15]. Table 1 summarizes the statistics and timings of our examples.

Comparison to Other Methods. Fig. 7 shows a comparison of the buckling effect between our method and state-of-the-art method [7]. We can notice that when only taking one global iteration, the surface details of both fluids can be quickly blurred. Then, as the global iteration number increases, significant improvement on preserving surface details can be found with our method. However, if we solve incompressibility with DFSPH and viscosity with an implicit viscosity (IV) solver multiple times, a side effect can be observed. The material becomes stiffer with more global

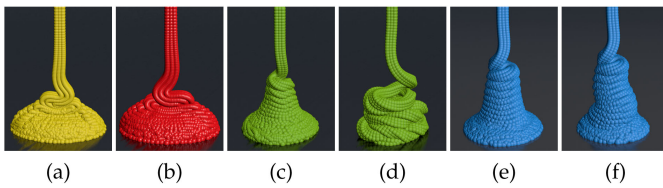


Fig. 8. Comparison of rope coiling effect. (a) Peer *et al.* [6]; (b) Peer *et al.* [13]; (c) Weiler *et al.* [7] with 1 iteration; (d) Weiler *et al.* [7] with 5 iterations; (e) Our method with 1 iteration; (f) Our method with 5 iterations.

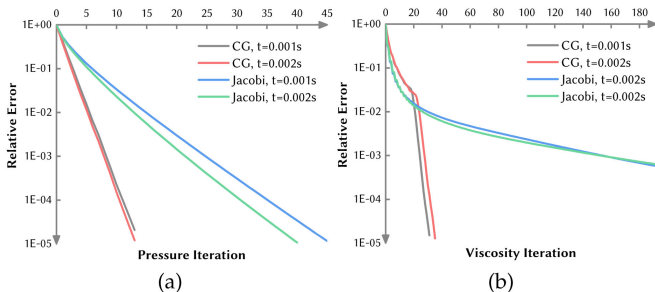


Fig. 9. Sub-iterative convergence analysis for the rope coiling example. (a) Convergence rates of the incompressibility solver; (b) Convergence rates of the viscosity solver.

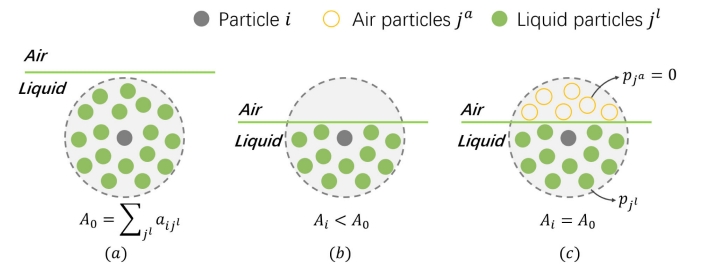


Fig. 10. The semi-analytical ghost particles. (a) An interior particle whose support domain is not clumped by boundary; (b) A boundary particle whose support domain intersects with the boundary; (c) A boundary particle with both ghost-air particles and liquid particles in its neighborhood.

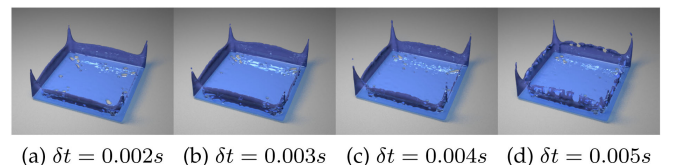


Fig. 11. A dambreak case consisting of 20k particles with a sampling distance of 0.005m. The support radius is set to 0.0125m. We can notice consistent behaviors of simulations that are taken under four different time steps.

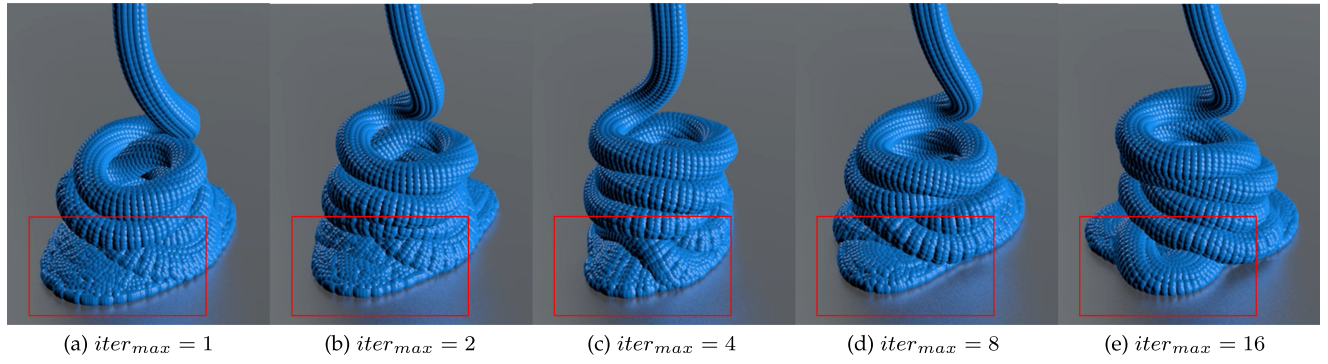


Fig. 12. Rope coiling effect for a fluid with a viscosity of $\mu = 5000 \text{ Pa} \cdot \text{s}$. By taking more iterations for the outer loop, our method is able to preserve more sharp surface details without losing viscous behaviors. The radiuses of the rope coiling effects are almost the same, which means that the physical viscosities do not change with the iteration number [45].

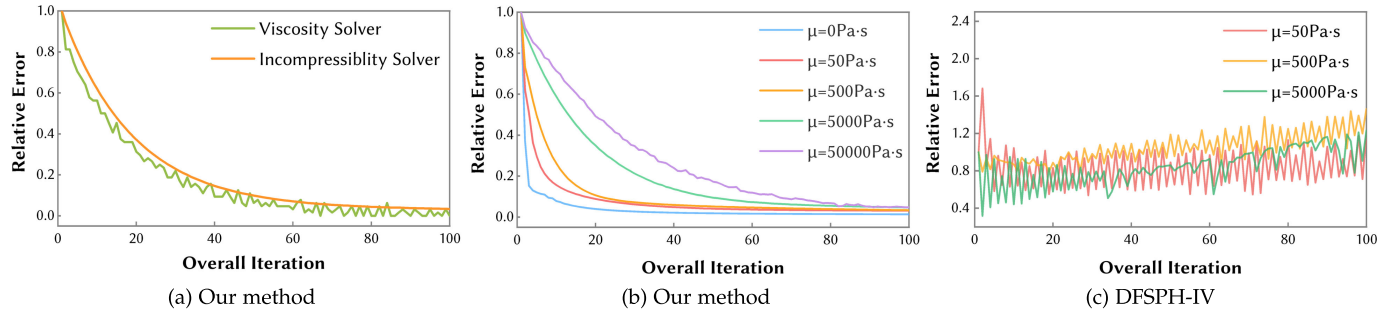


Fig. 13. Evaluation of the global convergence rate. (a) In simulating an incompressible fluid with a viscosity of $\mu = 5000 \text{ Pa} \cdot \text{s}$, our method shows that the relative errors of both the viscosity and incompressibility solvers converge to zero synchronously as the number of global iteration increases. (b) Comparison of the convergence rates of our method in simulating fluids with different viscosity coefficients, note the global convergence rate depends mostly on the viscosity coefficient. (c) Instead, if we use DFSPH to solve fluid incompressibility, the algorithm fails to converge to a global solution. In all above tests, the thresholds for both the viscous and incompressible solvers are set to 10^{-5} .

iterations (please count the wrinkle number). It indicates the combination of DFSPH and the viscosity solver does not converge to a global solution. The cause of this problem is that if incompressibility and viscosity solvers are not linked properly (as discussion in Section 3), the final solution could deviate from the global solution. In contrast, our method does not suffer from this problem due to the linkage between pressure and shear forces.

Since earlier works [6], [13] have also been denoted to remove the interference between pressure and viscosity, we provide a more thorough comparison in Fig. 8. As shown in Figs. 8a and 8b, both the methods of Peer *et al.* 2015 [6] and Peer *et al.* 2016 [13] fail to capture the coiling effect correctly. This problem is caused by the missing of rotational motion due to the damping introduced by their reconstruction of the velocity field [7]. With Weiler *et al.* [7]'s method, the correct rope coiling effect is visible in Fig. 8c. However, if we want to enrich surface details by taking 5 SIMPLE iterations, Fig. 8d shows the correct coiling effect is damaged. In other words, the DFSPH+IV strategy does not guarantee the convergence under multiple SIMPLE iterations. In contrast, Figs. 8e and 8f show that our method guarantees a consistent behavior with more surface details when more iterations are taken.

Newtonian Fluids. A Newtonian fluid is a fluid whose viscous stress is linearly proportional to the local strain rate, i.e., μ is a constant. To show that our method is stable at simulating Newtonian fluids with any viscosity, we drop a bunny of liquid with three different viscosities on a bar, as shown in Fig. 1. The corresponding viscosity coefficients are

set to $0 \text{ Pa} \cdot \text{s}$, $300 \text{ Pa} \cdot \text{s}$ and an extremely high value of $5 \times 10^7 \text{ Pa} \cdot \text{s}$, respectively. Note for the non-viscous fluid, only one outer loop iteration is needed because no viscous forces exist to violate fluid incompressibility.

Non-Newtonian Fluids. In non-Newtonian fluids, viscosity can change with varying shear rates. We implement the commonly used Cross model [50] to simulate non-Newtonian fluids. Its dynamic viscosity is defined as the following nonlinear function

$$\mu = \mu_{\infty} + \frac{\mu_0 - \mu_{\infty}}{1 + (k\dot{\gamma})^n}, \quad (22)$$

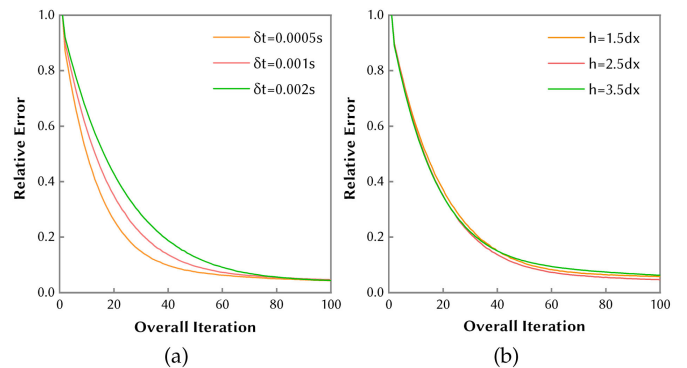


Fig. 14. Evaluation of the influence of the time step size (δt) and particle sampling distance (dx) on the global convergence rate for the rope coiling example ($\mu = 5000 \text{ Pa} \cdot \text{s}$, $\epsilon_0^v = 10^{-5}$, $\epsilon_0^p = 10^{-5}$). (a) Global convergence rates of the incompressibility solver with different time step sizes; (b) Global convergence rates of the incompressibility solver with different particle sampling distances.

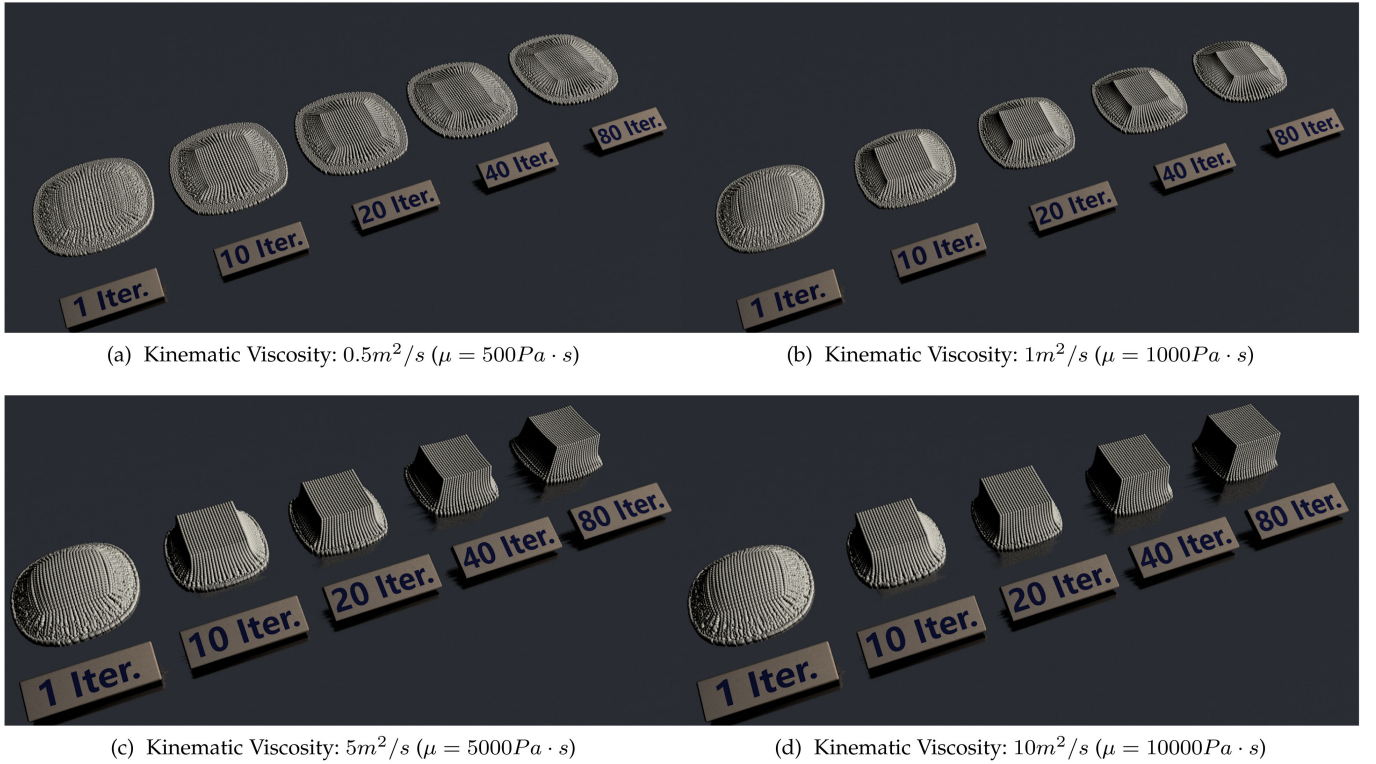


Fig. 15. Evaluation of the dependence of viscous behaviors on the SIMPLE iteration number. The figure shows a snapshot of the viscous fluids 2.5 seconds after they first hit the ground. In this scenario, each fluid block consists of 8k particles and the corresponding SIMPLE iteration number is displayed on the board. A free-slip boundary condition is imposed on the ground. The surface tension is neglected to make surface details more obvious. Stable fluid patterns are reached as the iteration number is increased.

TABLE 1
Statistics and Timings per Timestep of Our Examples

Name	total particles	$\mu (Pa \cdot s)$	surface tension	$\delta t [ms]$	$iter_{max}$	$t_p [ms]$	$t_v [ms]$	$t_{total} [s]$
Fig. 1a	36k	0	0	1.0	1	5.5	0	3.2×10^{-2}
Fig. 1b	36k	300	0	1.0	2	5.5	4.4	5.1×10^{-2}
Fig. 1c	36k	5×10^7	0	1.0	2	5.5	13.1	7.0×10^{-2}
Fig. 2	48k	dynamic	0	2.0~3.0(CFL)	5	6.3	9.1	0.15
Fig. 3	136k	3, 100, 5×10^5	0	1.0	2	31	49	0.27
Fig. 4	462k	400	0.002	1.4~2.2(CFL)	2	65	135	1.6
Fig. 7d	36k	5000	0	1.0	1	5.3	8.2	6.1×10^{-2}
Fig. 7e	36k	5000	0	1.0	5	5.3	8.2	0.11
Fig. 7f	36k	5000	0	1.0	10	5.3	8.2	0.18
Fig. 8e	5k	5000	0	1.0	1	1.0	1.7	1.1×10^{-2}
Fig. 8f	5k	5000	0	1.0	5	1.0	1.7	2.0×10^{-2}
Fig. 12a	22k	5000	0.0005	1.0	1	4.1	5.2	3.6×10^{-2}
Fig. 12b	22k	5000	0.0005	1.0	2	4.1	5.2	4.5×10^{-2}
Fig. 12c	22k	5000	0.0005	1.0	4	4.1	5.2	6.4×10^{-2}
Fig. 12d	22k	5000	0.0005	1.0	8	4.1	5.2	0.11
Fig. 12e	22k	5000	0.0005	1.0	16	4.1	5.2	0.19
Fig. 16(left)	104k	5×10^6	0	1.0	1	16	37	0.13
Fig. 16(right)	104k	5×10^6	0	1.0	5	16	37	0.34
Fig. 17a	1068k	200, 300, 5×10^4 , 5×10^5	0, 0.0005	1.0	5	90	203	2.4
Fig. 17b	54k	5×10^7	0	1.0	5	7.1	10.2	0.15
Fig. 18	975k	5×10^4	0	1.2~1.9(CFL)	4	82	171	1.9
Fig. 19	103k	dynamic	0	1.0	2	17	41	0.38

$iter_{max}$ represents the number of SIMPLE iteration, t_p represents the average computational cost for solving Equation (7), t_v represents the average computational cost for solving Equation (5), t_{total} represents the average computational cost per time step.



Fig. 16. Creams on cookies. Compared to a one-iteration strategy on the left, we can notice more surface details on the right when five SIMPLE iterations are taken.

where μ_0 and μ_∞ are two limiting values of viscosity at very low and high shear rates, k controls the shear rate and n controls the smoothness of transition, $\dot{\gamma}$ is the shear rate defined as

$$\dot{\gamma} = \sqrt{0.5 \cdot \text{trace}(\mathbf{D})^2}, \quad \mathbf{D} = \nabla \mathbf{v} + \nabla \mathbf{v}^T. \quad (23)$$

To guarantee momentum conservation, the viscosity term for non-Newtonian fluids should be corrected as follows:

$$\mu \nabla^2 \mathbf{v}' = 2(d+2) \sum_{j,l} V_{jl} \frac{\mu_i + \mu_{jl}}{2} \frac{\mathbf{v}'_{ijl} \cdot \mathbf{r}_{ijl}}{|\mathbf{r}_{ijl}|^2 + \eta^2} \nabla W_{ijl}. \quad (24)$$

Fig. 2 shows both shear thinning ($\mu_0 = 50 \text{ Pa} \cdot \text{s}$, $\mu_\infty = 5 \times 10^5 \text{ Pa} \cdot \text{s}$, $n = -6$, $k = 10$) and shear thickening ($\mu_0 = 50 \text{ Pa} \cdot \text{s}$, $\mu_\infty = 5 \times 10^5 \text{ Pa} \cdot \text{s}$, $n = 6$, $k = 1$) behaviors simulated with the Cross viscosity model.

For comparison, a low viscosity as well as a high viscosity Newtonian fluid are also demonstrated. In this example, the two non-Newtonian fluids show dramatic change of the viscosity during their impact. The shear thickening fluid seems to be quickly hardened during the impact due to the increased viscosity and then softened gradually as the deformation slows down. In contrast, the shear thinning fluid crumbles as a result of material softening during the impact.

To further demonstrate the distinctive behaviors between Newtonian and non-Newtonian fluids, we setup another experiment for comparison, i.e., by dropping rigid objects

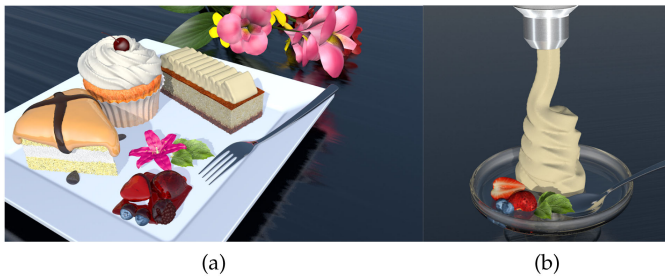


Fig. 17. Desserts and ice-cream. (a) Desserts. Creams with four different viscosities (maple cream: $\mu = 300 \text{ Pa} \cdot \text{s}$; chocolate cream and strawberry jam: $\mu = 200 \text{ Pa} \cdot \text{s}$; faint yellow cream: $\mu = 5 \times 10^4 \text{ Pa} \cdot \text{s}$; white cream: $\mu = 5 \times 10^5 \text{ Pa} \cdot \text{s}$) are simulated. (b) Ice-cream. Our method captures the rope coiling effect of the ice-cream without losing surface details when the interference between pressure and shear forces is alleviated.

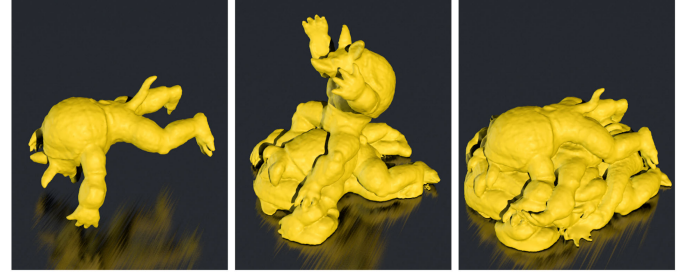


Fig. 18. Five armadillos falling onto the ground. This example shows that our method preserves shapes and surface details well for a large scale simulation of a high viscosity fluid.

into Newtonian fluids and non-Newtonian fluids respectively, as shown in Fig. 19. Both the low viscosity Newtonian fluid and shear thickening fluid are simulated with an initial viscosity of $\mu = 5 \text{ Pa} \cdot \text{s}$. However, as the cube touches the liquid surface, the rapid change of deformation in the shear thickening fluid helps to increase shear forces, therefore slows the motion of the cube quickly. In contrast, when the simulation of a high viscosity Newtonian fluid and a shear-thinning fluid are started with an initial viscosity of $\mu = 5 \times 10^5 \text{ Pa} \cdot \text{s}$, the shear thinning fluid softens as the cube touch the liquid surface due to the rapid change of deformation. Note that for a shear thinning fluid, the value of $\dot{\gamma}$ could occasionally be 0. To avoid being divided by zero, we simply neglect the last term of Equation (22) when $\dot{\gamma}$ is smaller than a predefined threshold.

Cream on Cookies. To demonstrate the effectiveness of our method in removing the interference between pressure and shear forces, Fig. 16 demonstrates a comparison of two simulations where on the left only one SIMPLE iteration is taken to simulate the cream while 5 iterations are taken on the right. According to the comparison, we can notice more surface details with more iterations. Besides, taking more iterations also helps preserve the final shape for a longer time.

More Demonstrations. In Fig. 3, three bunnies with different viscosities are dropped onto a table. Different viscous behaviors as well as interactions between them can be noticed. In Fig. 18, five armadillos are dropped sequentially onto the ground. This example shows that our method preserves shapes and surface details well for a high viscosity fluid ($\mu = 5 \times 10^4 \text{ Pa} \cdot \text{s}$). Fig. 4 shows an armadillo consisting of 462k particles dropped onto a bench. By including

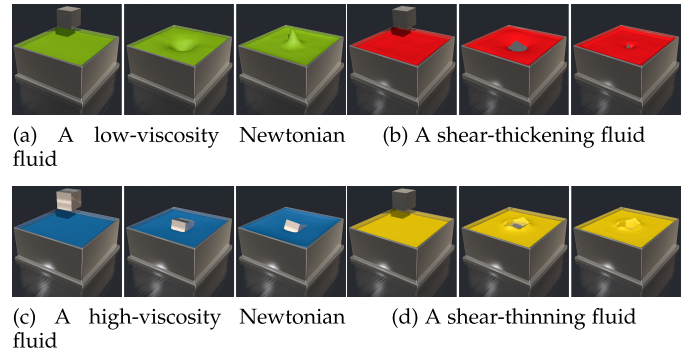


Fig. 19. Rigid objects dropped into Newtonian and non-Newtonian fluids. (a) A low viscosity Newtonian fluid with $\mu = 5 \text{ Pa} \cdot \text{s}$; (b) A shear thickening fluid with $\mu_0 = 5 \text{ Pa} \cdot \text{s}$, $\mu_\infty = 5 \times 10^5 \text{ Pa} \cdot \text{s}$, $n = 6$ and $k = 1$; (c) A high viscosity Newtonian fluid with $\mu = 5 \times 10^5 \text{ Pa} \cdot \text{s}$; (d) A shear thinning fluid with $\mu_0 = 5 \text{ Pa} \cdot \text{s}$, $\mu_\infty = 5 \times 10^5 \text{ Pa} \cdot \text{s}$, $n = -10$ and $k = 50$.

surface tension forces [42], our method is able to reproduce remarkable viscous behaviors, such as rope coiling and buckling, as well as other realistic small-scale thin features, such as filaments and sheets. In Fig. 17b, a real scenario of making ice cream is simulated to demonstrate the rope coiling effect ($\mu = 5 \times 10^4 \text{ Pa} \cdot \text{s}$). Due to our SIMPLE method, the ice cream is able to preserve the shape well. In Fig. 17a, four different Newtonian fluids were simulated by our method. The maple cream on the triangle cake is simulated with a viscosity of $300 \text{ Pa} \cdot \text{s}$ and the chocolate cream with $\mu = 200 \text{ Pa} \cdot \text{s}$. The faint yellow cream ($\mu = 5 \times 10^4 \text{ Pa} \cdot \text{s}$, 5 SIMPLE iterations) is dropped onto a square cake to demonstrate the buckling effect. The white cream ($\mu = 5 \times 10^5 \text{ Pa} \cdot \text{s}$, 5 SIMPLE iterations) dropped onto the round cake demonstrates the rope coiling effect.

Limitations. The major limitation of our method is the high computational overhead because the total computation cost is linearly proportional to iteration number of the outer loop. Besides, the convergence rate to a global solution depends on the viscosity coefficient. However, in most cases, a small fixed number of SIMPLE iterations will be enough to achieve desirable visual effects. Another strategy to improve the performance is to decrease the iteration number of inner loops. This works because the incompressibility and viscosity solvers actually deteriorate the accuracy between each other. The final limitation is that the discretized pressure gradient in Equation (13) does not strictly conserve the total linear momentum. However, during our tests, no apparent visual artifacts have been observed.

6 CONCLUSION

In this paper, we presented a SIMPLE algorithm adapted for SPH in simulating viscous incompressible fluids. The main purpose is to alleviate the interference between pressure and shear forces, especially for highly viscous incompressible fluids. We have demonstrated a variety of different examples to show the effectiveness of our method in alleviating the interference artifact. Our method is able to model remarkable viscous behaviors, such as coiling and buckling, with a good property of preserving sharp surface details. Besides, our method is stable at simulating incompressible Newtonian and non-Newtonian fluids with any given viscosity.

For our further work, we will first consider how to improve the performance. Besides, we will consider extending our method to simulate more kinds of viscous incompressible fluids, especially for those with thin features. Finally, we will investigate how to alleviate the interference between viscoelasticity and incompressibility for viscoelastic incompressible fluids.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable comments. The work was supported by the National Key R&D Program of China under Grant 2017YFB1002701, the National Natural Science Foundation of China under Grants 62072446, 61872345, 61672502, 62072449, 61632003, the Youth Innovation Promotion Association, CAS under Grant 2019109, the Strategic Priority Research Program, CAS under Grant 19080102, and the University of Macau under Grant MYRG2019-00006-FST.

REFERENCES

- [1] B. Solenthaler and R. Pajarola, "Predictive-corrective incompressible SPH," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 40:1–40:6, Jul. 2009.
- [2] M. Ihmsen, J. Cornelis, B. Solenthaler, C. Horvath, and M. Teschner, "Implicit incompressible SPH," *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 3, pp. 426–435, Mar. 2014.
- [3] J. J. Monaghan, "Smoothed particle hydrodynamics," *Rep. Progress Phys.*, vol. 68, no. 8, pp. 1703–1759, Jul. 2005.
- [4] M. Becker and M. Teschner, "Weakly compressible SPH for free surface flows," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2007, pp. 209–217.
- [5] T. Takahashi, Y. Dobashi, I. Fujishiro, T. Nishita, and M. C. Lin, "Implicit formulation for SPH-based viscous fluids," *Comput. Graph. Forum*, vol. 34, no. 2, pp. 493–502, 2015.
- [6] A. Peer, M. Ihmsen, J. Cornelis, and M. Teschner, "An implicit viscosity formulation for SPH fluids," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 114:1–114:10, Jul. 2015.
- [7] M. Weiler, D. Koschier, M. Brand, and J. Bender, "A physically consistent implicit viscosity solver for SPH fluids," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 145–155, 2018.
- [8] E. Larionov, C. Batty, and R. Bridson, "Variational stokes: A unified pressure-viscosity solver for accurate viscous liquids," *ACM Trans. Graph.*, vol. 36, no. 4, 2017, Art. no. 101.
- [9] S. Shao and E. Y. Lo, "Incompressible SPH method for simulating newtonian and non-newtonian flows with a free surface," *Advances Water Resour.*, vol. 26, no. 7, pp. 787–800, 2003.
- [10] X. He *et al.*, "Staggered meshless solid-fluid coupling," *ACM Trans. Graph.*, vol. 31, no. 6, 2012, Art. no. 149.
- [11] C. Gissler, A. Peer, S. Band, J. Bender, and M. Teschner, "Interlinked SPH pressure solvers for strong fluid-rigid coupling," *ACM Trans. Graph.*, vol. 38, no. 1, 2019, Art. no. 5.
- [12] J. Bender and D. Koschier, "Divergence-free smoothed particle hydrodynamics," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2015, pp. 147–155.
- [13] A. Peer and M. Teschner, "Prescribed velocity gradients for highly viscous SPH fluids with vorticity diffusion," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 12, pp. 2656–2662, Dec. 2017.
- [14] S. Patankar, *Numerical Heat Transfer and Fluid Flow*. Boca Raton, FL, USA: CRC Press, 2018.
- [15] J. J. Monaghan, "Smoothed particle hydrodynamics," *Annu. Rev. Astron. Astrophys.*, vol. 30, no. 1, pp. 543–574, 1992.
- [16] S. J. Cummins and M. Rudman, "An SPH projection method," *J. Comput. Phys.*, vol. 152, no. 2, pp. 584–607, 1999.
- [17] K. Bodin, C. Lacoursiere, and M. Servin, "Constraint fluids," *IEEE Trans. Vis. Comput. Graphics*, vol. 18, no. 3, pp. 516–526, Mar. 2012.
- [18] N. Akinci, M. Ihmsen, G. Akinci, B. Solenthaler, and M. Teschner, "Versatile rigid-fluid coupling for incompressible SPH," *ACM Trans. Graph.*, vol. 31, no. 4, 2012, Art. no. 62.
- [19] J. J. Monaghan, "On the problem of penetration in particle methods," *J. Comput. Phys.*, vol. 82, pp. 1–15, 1989.
- [20] H. Schechter and R. Bridson, "Ghost SPH for animating water," *ACM Trans. Graph.*, vol. 31, no. 4, 2012, Art. no. 61.
- [21] M. Macklin and M. Müller, "Position based fluids," *ACM Trans. Graph.*, vol. 32, no. 4, 2013, Art. no. 104.
- [22] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2003, pp. 154–159.
- [23] B. Solenthaler and R. Pajarola, "Density contrast SPH interfaces," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2008, pp. 211–218.
- [24] B. Jones, S. Ward, A. Jallepalli, J. Perenia, and A. W. Bargteil, "Deformation embedding for point-based elastoplastic simulation," *ACM Trans. Graph.*, vol. 33, no. 2, 2014, Art. no. 21.
- [25] R. Winchenbach, H. Hochstetter, and A. Kolb, "Infinite continuous adaptivity for incompressible SPH," *ACM Trans. Graph.*, vol. 36, no. 4, 2017, Art. no. 102.
- [26] J. Monaghan, "Simulating free surface flows with SPH," *J. Comput. Phys.*, vol. 110, no. 2, pp. 399–406, 1994.
- [27] J. P. Morris, P. J. Fox, and Y. Zhu, "Modeling low Reynolds number incompressible flows using SPH," *J. Comput. Phys.*, vol. 136, no. 1, pp. 214–226, 1997.
- [28] X. He, N. Liu, S. Li, H. Wang, and G. Wang, "Local poisson SPH for viscous incompressible fluids," *Comput. Graph. Forum*, vol. 31, no. 6, pp. 1948–1958, 2012.
- [29] J. Cornelis, J. Bender, C. Gissler, M. Ihmsen, and M. Teschner, "An optimized source term formulation for incompressible SPH," *Vis. Comput.*, vol. 35, pp. 579–590, 2019.

- [30] M. Ihmsen, J. Orthmann, B. Solenthaler, A. Kolb, and M. Teschner, "SPH fluids in computer graphics," *Eurographics Assoc.*, 2014.
- [31] H. Gotoh and A. Khayyer, "Current achievements and future perspectives for projection-based particle methods with applications in ocean engineering," *J. Ocean Eng. Marine Energy*, vol. 2, no. 3, pp. 251–278, 2016.
- [32] D. Koschier, J. Bender, B. Solenthaler, and M. Teschner, "Smoothed particle hydrodynamics for physically-based simulation of fluids and solids," *Eurographics Assoc.*, 2019.
- [33] C. Batty, F. Bertails, and R. Bridson, "A fast variational framework for accurate solid-fluid coupling," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 100:1–100:7, Jul. 2007.
- [34] M. Aanjaneya, S. Patkar, and R. Fedkiw, "A monolithic mass tracking formulation for bubbles in incompressible flow," *J. Comput. Phys.*, vol. 247, no. 2, pp. 17–61, 2013.
- [35] A. Robinson-Mosher, T. Shinar, J. Gretarsson, J. Su, and R. Fedkiw, "Two-way coupling of fluids to rigid and deformable solids and shells," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–9, 2008.
- [36] A. Robinson-Mosher, C. Schroeder, and R. Fedkiw, "A symmetric positive definite formulation for monolithic fluid structure interaction," *J. Comput. Phys.*, vol. 230, no. 4, pp. 1547–1566, 2011.
- [37] M. Liu and G. Liu, "Smoothed particle hydrodynamics (SPH): An overview and recent developments," *Arch. Comput. Methods Eng.*, vol. 17, no. 1, pp. 25–76, 2010.
- [38] J.-P. Fürstenu, B. Avci, and P. Wriggers, "A comparative numerical study of pressure-poisson-equation discretization strategies for SPH," in *Proc. 12th Int. SPHERIC Workshop*, 2017, pp. 1–8.
- [39] S. Yang et al., "Enriching SPH simulation by approximate capillary waves," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2016, pp. 29–36.
- [40] J. O. Cruickshank and B. R. Munson, "Viscous fluid buckling of plane and axisymmetric jets," *J. Fluid Mech.*, vol. 113, pp. 221–239, 1981.
- [41] P. Nair and G. Tomar, "An improved free surface modeling for incompressible SPH," *Comput. Fluids*, vol. 102, pp. 304–314, 2014.
- [42] X. He, H. Wang, F. Zhang, H. Wang, G. Wang, and K. Zhou, "Robust simulation of sparsely sampled thin features in SPH-based free surface flows," *ACM Trans. Graph.*, vol. 34, no. 1, pp. 7:1–7:9, Dec. 2014.
- [43] P. Nair and G. Tomar, "An improved free surface modeling for incompressible SPH," *Comput. Fluids*, vol. 102, pp. 304–314, 2014.
- [44] T. Takahashi, Y. Dobashi, T. Nishita, and M. C. Lin, "An efficient hybrid incompressible SPH solver with interface handling for boundary conditions," *Comput. Graph. Forum*, vol. 37, no. 1, pp. 313–324, 2018.
- [45] M. Habibi, M. Maleki, R. Golestanian, N. M. Ribe, and D. Bonn, "Dynamics of liquid rope coiling," *Phys. Rev. E*, vol. 74, 2006, Art. no. 066306.
- [46] F. Losasso, J. Talton, N. Kwatra, and R. Fedkiw, "Two-way coupled SPH and particle level set fluid simulation," *IEEE Trans. Vis. Comput. Graphics*, vol. 14, no. 4, pp. 797–804, Jul./Aug. 2008.
- [47] A. Khayyer and H. Gotoh, "Enhancement of stability and accuracy of the moving particle semi-implicit method," *J. Comput. Phys.*, vol. 230, no. 8, pp. 3093–3118, 2011.
- [48] X. He, H. Wang, G. Wang, H. Wang, and E. Wu, "A variational staggered particle framework for incompressible free-surface flows," 2020, *arXiv: 2001.09421*.
- [49] H. Bhattacharya, Y. Gao, and A. W. Bargteil, "A level-set method for skinning animated particle data," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2011, pp. 17–24.
- [50] M. M. Cross, "Rheology of non-newtonian fluids: A new flow equation for pseudoplastic systems," *J. Colloid Sci.*, vol. 20, no. 5, pp. 417–437, 1965.



Shusen Liu received the BS degree from Heilongjiang University, Harbin, China, in 2011, and the MS degree from Jilin University, Changchun, China, in 2014. Currently, he is working toward the PhD degree with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests include computer graphics and physical-based simulation.



Xiaowei He received the BS and MS degrees from Peking University, Beijing, China, in 2008 and 2011, respectively, and the PhD degree from the Institute of Software, Chinese Academy of Science(CAS), Beijing, China, in 2017. He is currently a research associate professor with the HCI Lab of Institute of Software, Chinese Academy of Science. His research interests include computer graphics and physical-based simulation.



Wencheng Wang (Member, IEEE) received the PhD degree in software from the Institute of Software, Chinese Academy of Sciences, Beijing, China, in 1998. He is currently a professor with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, where he leads a Research Group on Computer Graphics and Image Processing. His research interests include computational geometry, computer graphics, visualization, and image editing. He is a member of the ACM.



Enhua Wu (Member, IEEE) received the BS degree from Tsinghua University, Beijing, China, and the PhD degree from the University of Manchester, Manchester, U.K., in 1984. He is currently a research professor with the State Key Laboratory of Computer Science, Chinese Academy of Sciences since 1985. He has also been teaching with the University of Macau since 1997, where he is currently an emeritus professor, and the dean of ZhuhaiUM Science and Technology Research Institute. He has been invited to chair or co-chair various conferences or program committees including in recent years SIGGRAPH Asia 2016, ACM VRST2010, 2013, 2015, WSCG2012. He is an associate editor-in-chief of the *Journal of Computer Science and Technology*, and the associate editor of the *The Visual Computer*, *Computer Animation and Virtual Worlds*, *International Journal of Image and Graphics*, *International Journal of Software and Informatics*. His research interests include realistic image synthesis, physically based simulation, and virtual reality. He is a member of the ACM, and a fellow of the China Computer Federation (CCF).

► For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.