Incompressibility Enforcement for Multiple-fluid SPH Using Deformation Gradient

Bo Ren, *Member, IEEE,* Wei He, Chen-feng Li, and Xu Chen

Abstract—To maintain incompressibility in SPH fluid simulations is important for visual plausibility. However, it remains an outstanding challenge to enforce incompressibility in such recent multiple-fluid simulators as the mixture-model SPH framework. To tackle this problem, we propose a novel incompressible SPH solver, where the compressibility of fluid is directly measured by the deformation gradient. By disconnecting the incompressibility of fluid from the conditions of constant density and divergence-free velocity, the new incompressible SPH solver is applicable to both single- and multiple-fluid simulations. The proposed algorithm can be readily integrated into existing incompressible SPH frameworks developed for single-fluid, and is fully parallelizable on GPU. Applied to multiple-fluid simulations, the new incompressible SPH scheme significantly improves the visual effects of the mixture-model simulation, and it also allows exploitation for artistic controlling.

Index Terms—compressible and incompressible fluids, multiple-fluid simulation, Smoothed Particle Hydrodynamics, deformation gradient

1 Introduction

The Smoothed-Particle Hydrodynamics (SPH) method is popular in computer graphics for simulations of a wide range of liquid and solid motions. It is often convenient and desirable to treat real-world liquids and solids as incompressible in computer simulations. However, enforcing incompressibility in SPH simulation is harder than that in grid-based methods, because the meshless scheme features a constantly-changing neighbourhood of particles. In recent years, multiple-fluid SPH algorithms have emerged, further increasing the complexity of the problem. This paper proposes a generic approach to uniformly address the incompressibility issue in SPH fluid simulations.

There are already several literatures to tackle incompressibility in single-fluid SPH simulation. Through enforcing the constant density and/or the divergence-free velocity field, previous methods have achieved a high degree of incompressibility and good computational performance in single-fluid simulation. These techniques are readily applicable to certain multiple-fluid simulators using simplified models [1], [2], where all phases have an identical velocity such that the velocity field becomes divergence free. However, real world multiple-fluid mixtures are much more complex, with different phases flowing at different velocities. Thus, neither the divergence-free nor the constant-density assumption holds for real world multiple-fluid flows. In the fundamental theories of multiple-fluid flows, e.g. the mixture model [3], [4], although the densities of individual phases can be assumed as constants, the mixture contains several phases with varying fractions such that its rest density is not a constant and usually unknown. Within the typical SPH scheme where particle mass does not change after initialization, neither the phase velocity field nor the particle velocity field is divergence free for real world multiple-fluid scenarios. As a result, it is hard to extend the existing incompressible SPH schemes into real world multiple-fluid flows described by the mixture model. We observe that incompressibility is a physical property to describe a material resisting volume change caused by various forces. Such property always holds in incompressible flows, no matter how the fluid composition changes in complex multiple-fluid flows. For example, with the presence of chemical reaction, dissolution or phase transition, individual fluid phases may increase/decrease in mass and volume, resulting in intrinsic changes to the fluid density and the velocity divergence, but each single phase itself is still incompressible. In such cases "physical compressibility" does exist when we treat the multiple-fluid mixture as a whole even if we assume each individual phase is incompressible, i.e. they still resist force-induced volume changes. For a practical incompressible mixture-model simulator, a new method is required to eliminate only the non-physical compressibility. On the other hand, the physical compressibility directly relates to a change of the total material volume, which in turn can be measured based on the deformation gradient tensor, whose Jacobian gives the volume expansion ratio. The deformation gradient is a geometric concept originating from solid mechanics (e.g. [5]), which can be used to describe incompressible plastic solids. We integrate this concept in fluid simulation and derive a novel scheme to enforce incompressibility in SPH multiple-fluid simulations.

In this paper we derive a novel incompressibility enforcement approach which directly measures the physical and non-physical volume variations of fluids using the deformation gradient. The new incompressible SPH scheme is generic and applicable to both single- and multiple-fluid simulations. Specifically, our main goal is providing an

- *Bo Ren, Wei He and Xu Chen are with the College of Computer Science, Nankai University, Tianjin. E-mail: rb@nankai.edu.cn*
- *Chen-feng Li is with Swansea University, UK. E-mail: c.f.li@swansea.ac.uk*

incompressible solver that significantly improves the visual effects and computational efficiency for incompressible mixture-model multiple-fluid simulation [3], which does not rely on the strong assumption, that all phase velocities are the same, like in phase-field methods [1], [2]. For single-fluid simulation, its computational performance is comparable to previous methods such as IISPH [6]. The main contribution of this work includes:

- A generic incompressibility enforcement scheme is derived by using the deformation gradient to measure particle volume change in SPH fluid simulation.
- Based on the new incompressible SPH scheme, an incompressible mixture-model multiple-fluid solver is established. The scheme is also studied on single-fluid simulation showing comparable performance with previous methods.
- The new algorithm is compatible with existing simulation frameworks.

The rest of the paper is organized as follows. §2 summarizes related works, after which the fundamental idea of the new approach and the concept of deformation gradient are explained in §3. In §4, the new incompressible SPH scheme is formulated for multiple-fluid simulation based on the mixture model. The effectiveness and performance of the new incompressible SPH approach are examined in §5 with various examples.

2 Related Work

The incompressibility enforcement has long accompanied the development of SPH fluid solvers. In the early research works [7], [8], [9], the SPH fluid is solved in a weakly compressible manner dependent on a stiffness coefficient in the equation of state: a higher stiffness value corresponds to better incompressibility, but at the cost of smaller time steps. Since the time step quickly decreases as the stiffness coefficient grows, it is often too costly for computer graphics applications to ensure visually incompressible results simply by raising the stiffness value.

Inspired by the grid-based fluid solvers, pressure projection schemes are developed to enforce incompressibility in SPH fluid simulation. Early approaches use direct SPH discretization of the Laplacian [10], [11], which is sensitive to sampling and does not scale well. Background grid is introduced later to alleviate the sensitivity issues [12], [13], [14]. More recently, the convergence is greatly improved with a better discretization method in the implicit incompressible SPH (IISPH) [6], which is also friendly to GPU implementation [15].

Some research works use an iterative prediction-correction scheme to enhance the incompressibility of SPH fluid. The solver first predicts an intermediate state by advecting the particles, and iteratively solves for the velocity correction. This is done by calculating an update term on velocity relying on the pressure [16], the constraint force based on the rigid body mechanics theory [17], or the divergence-free velocity [18]. Direct particle displacement correction after the prediction is proposed in position based fluid [19] to accelerate the computation. By combining constant density and divergence-free velocity corrections, the divergence free SPH (DFSPH) method [20] further accelerates the convergence.

The aforementioned methods mainly cope with incompressible single-fluid simulation, and they do not work with multiple-fluid simulations that have multiple phases coexisting in a SPH particle. Recent particle-based multiple-fluid models include the mixture-model multi-fluid SPH solvers [3], [4], phase-field based multi-fluid SPH solvers [1], [2] and multi-phase MPM solvers for miscible flow [21] and sand-water mixtures [22], [23]. Among those methods, the mixture model solvers can reproduce layered unmixing effect such as centrifugal separating, but are hardest to enforce incompressibility because neither mixture density is constant nor the velocity fields are divergence free. To address this problem, we propose a generic incompressibility enforcement scheme that is applicable to both single- and multiple-fluid SPH simulations.

3 Deformation Gradient

This section introduces the deformation gradient and related formulation to enforce incompressibility in fluid simulation. For simplicity, we use single-fluid flow to demonstrate the derivation.

The single-fluid flow is governed by a continuity equation and a momentum equation as follows:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho}\nabla p + \frac{1}{\rho}\mathbf{M} \tag{2}$$

where $\rho, p, \mathbf{u}, \mathbf{M}$ denote density, pressure, velocity and non-pressure force of the fluid. In single-fluid simulation the incompressibility assumption usually refers to constant density, and consequently Eqn. (1) leads to the divergence-free condition for velocity, i.e. $\nabla \cdot \mathbf{u} = 0$. In grid-based solvers the divergence-free condition can be treated as a constraint in solving the momentum equation and is resolved in a projection step. However, in SPH simulation the particle neighbourhood is constantly changing, which makes the problem more complex. [6] studied this issue and proposed that better efficiency could be achieved by using known values from the current-frame particle neighborhood to enforce incompressibility than recomputing particle neighbourhood each time.

From a more fundamental perspective, the visual "compressibility" observed during a fluid simulation can come from two possible sources. First, the simulator may not accurately enforce the incompressibility due to a suboptimal model or numerical errors in the simulation. Secondly, the fluid itself may experience changes of physical properties resulting in volume expansion or shrinkage. In graphics applications the visual plausibility is mostly affected by the former source, i.e. non-physical compressibility. It is also noted that the latter source, i.e. physical compressibility, is fairly common in multiple-fluid flows where such processes as reaction, dissolution and phase transition can change the fluid composition and density. In multiple-fluid simulation, an incompressible solver should aim at eliminating the non-physical compressibility while allowing density change and

velocity divergence originated from physical changes of fluid. That is, the volume change of fluid phases must match the change of fluid composition.

Based on the above observation, we propose a novel incompressible SPH enforcement scheme using the deformation gradient tensor. The deformation gradient tensor can be computed from the current-frame particle topology avoiding re-computation of particle neighbourhoods, and it measures directly the volume variation of fluids. As a key concept in continuum mechanics, the deformation gradient quantifies the deformation of a continuum material and it is defined using the gradient of deformed positions measured under the original frame:

$$\mathbf{F} = (\nabla_0 \mathbf{x})^T = I + \Delta t (\nabla_0 \mathbf{u})^T \tag{3}$$

where $\mathbf{F}$ is the deformation gradient tensor, $\mathbf{x}$ and $\mathbf{u}$ are position and velocity of the deformed shape, ∇_0 represents the material gradient operator and the subscript 0 denotes the gradient with respect to the original frame. A similar equation as above can be found in [24]. The second equality is obtained with a linear local approximation between two adjacent time steps.

An important property of the deformation gradient is that its Jacobian $J = |\mathbf{F}| = \frac{V}{V_0}$ is a measure of the volume change produced by a deformation, where V_0 and V are volumes of the meta elements in the original and deformed frames, respectively, and $|\cdot|$ is the determinant of the inside matrix. Therefore, for the simple case of a fixed-density fluid, enforcing incompressibility for non-physical causes is equivalent to ensuring $J = 1$ during the simulation, which is calculated purely from geometric quantities of the fluid flow. For multiple fluid simulations, J provides a direct indicator for whether the local volume change of a particle matches its physical compressibility causes such as composition change. We derive our novel scheme based on the direct measure J in the following sections.

The concept of deformation gradient applies to any material that has continuum representation, whether it is solid or fluid, and Eqns. (3) do not rely on incompressibility assumptions. The readers are referred to [25] for a more thorough explanation on the deformation gradient and related theory.

4 Derivation

The mixture model [3] simulates multiple-fluid flows using drift velocities of different phases and can reproduce smooth layered un-mixing effects. However it has to be solved in a weakly compressible SPH (WCSPH) framework [9], and its incompressibility issue remains an outstanding challenge. In this section we show how the deformation gradient and its Jacobian can be used to address the incompressibility issue in multiple-fluid simulations. The formulations are first derived for the single-fluid simulation to demonstrate the fundamental idea and the framework, then they are extended to multiple-fluid simulations for the mixture model.

4.1 Single-fluid Solver

As with previous methods, the intermediate particle position $\mathbf{x}^{adv}$, velocity $\mathbf{v}^{adv}$ and deformation gradient $\mathbf{F}^{adv}$ are first predicted by advecting the particle one step forward using known current-frame values. Then we calculate a pressure correction Δp that corrects the intermediate values so that J satisfies the incompressibility requirement. Details are given below.

From the momentum equation Eqn.(2) and SPH velocity update scheme, Δp will cause a velocity change $\mathbf{u}^p$:

$$\mathbf{u}^p = \Delta t \frac{\mathbf{\Gamma}^{\Delta p}}{m} \tag{4}$$

where for each particle i,

$$\mathbf{\Gamma}_i^{\Delta p} = -m_i \Sigma_{j \in NF(i)} m_j \left(\frac{\Delta p_i}{\tilde{\rho_i}^2} + \frac{\Delta p_j}{\tilde{\rho_j}^2} \right) \nabla W_{ij} \tag{5}$$

$NF(i)$ represents the particle i's neighborhood, $\tilde{\rho}$ is interpolated density, and W is the kernel function. This velocity change $\mathbf{u}^p$ will in turn results in a change of $\mathbf{F}$ defined in Eqn.(3):

$$\mathrm{d}\mathbf{F}^{adv} = \Delta t (\nabla_0 \mathbf{u}^p)^T \tag{6}$$

With constant particle mass $J = \frac{V_{t+1}}{V_t} = \frac{\rho_t}{\rho_{t+1}}$. The target density can be set as $\rho_{t+1} = \rho_0$ or other changeable values, and the following relation holds:

$$\frac{\tilde{\rho_i}}{\rho_0} = J = |\mathbf{F}^{adv} + \mathrm{d}\mathbf{F}^{adv}| \tag{7}$$

Here we use the interpolated density $\tilde{\rho_i}$ to reflect the current fluid density state in the simulator and set the target density $\rho_{t+1} = \rho_0$. In all later derivations, a term is by default calculated using values at time step t and we drop the subscript t unless for clarity. Any term that represents a value at time step $t+1$ will explicitly have a $t+1$ subscript. The right hand side of Eqn. (7) can be calculated using the Jacobi's formula

$$J = |\mathbf{F}^{adv}| + \mathrm{d}|\mathbf{F}^{adv}| = |\mathbf{F}^{adv}| + \mathrm{tr}(\mathbf{F}^{adv*} \mathrm{d}\mathbf{F}^{adv}) \tag{8}$$

where $\mathbf{F}^{adv*}$ is the adjugate matrix of $\mathbf{F}^{adv}$. We adopt the SPH discretization to compute $d\mathbf{F}^{adv} = \frac{\Delta t}{\tilde{\rho_i}} \mathbf{N}_i$, where $\mathbf{N}_i$ is a 3×3 matrix and the element in its X-th row and Y-th column is given by:

$$\mathbf{N}_i^{XY} = \Sigma_{j \in NF(i)} m_j (\mathbf{u}_{ji}^p)_X (\nabla W_{ij})_Y \tag{9}$$

where for a physical quantity Q, $Q_{ji} = Q_j - Q_i$, $(\mathbf{u}_{ji}^p)_X$ and $(\nabla W_{ij})_Y$ refer to the X, Y-th element in the vectors, respectively. Following Eqns. (6-9), for each particle i, we can obtain a linear equation with respect to velocity:

$$-(\frac{\tilde{\rho_i}}{\rho_0} - |\mathbf{F}_i^{adv}|)\tilde{\rho_i} = \Delta t \Sigma_{j \in NF(i)} m_j (\mathbf{u}_{ij}^p)^T (\mathbf{F}_i^{adv*})^T \nabla W_{ij} \tag{10}$$

Based on the above equation, we can further substitute Δp for $\mathbf{u}^p$. For convenience, we first rewrite

$$\begin{aligned} \Delta t^2 \frac{\mathbf{\Gamma}_i^p}{m_i} &= \underbrace{(-\frac{\Delta t^2}{\tilde{\rho_i}^2} \sum_{j \in \mathrm{NF(i)}} m_j \nabla W_{ij})}_{\mathbf{d}_{ii}} \Delta p_i + \underbrace{\sum_{j \in \mathrm{NF(i)}} (-\Delta t^2 \frac{m_j}{\tilde{\rho_j}^2} \nabla W_{ij})}_{\mathbf{d}_{ij}} \Delta p_j \\ &= \mathbf{d}_{ii} \Delta p_i + \sum_{j \in \mathrm{NF(i)}} \mathbf{d}_{ij} \Delta p_j \end{aligned} \tag{11}$$

Substituting Eqn. (11) into Eqn. (10) and after some mathematical derivation, we can obtain the following linear equation with respect to Δp:

$$
\begin{aligned}
&-(\frac{\tilde{\rho_i}}{\rho_0}-|\mathbf{F}_i^{adv}|)\tilde{\rho_i} = \underbrace{(\sum_{j\in \mathrm{NF(i)}} m_j(\mathbf{d}_{ii}-\mathbf{d}_{ji})^T(\mathbf{F}_i^{adv*})^T\nabla W_{ij})}_{a_{ii}}\Delta p_i \\
&+\underbrace{\sum_{j\in \mathrm{NF(i)}} m_j(\sum_{k\in \mathrm{NF(i)}} \mathbf{d}_{ik}\Delta p_k - \mathbf{d}_{jj}\Delta p_j - \sum_{\substack{l\in \mathrm{NF(j)} \\ -\{i\}}} \mathbf{d}_{jl}\Delta p_l)^T(\mathbf{F}_i^{adv*})^T\nabla W_{ij}}_{dterm(\Delta p)} \\
&= a_{ii}\Delta p_i + dterm(\Delta p) \qquad (12)
\end{aligned}
$$

The same relaxed Jacobi method as in IISPH [6] can be used to iteratively solve for Δp:

$$\Delta p_i^{s+1} = (1-\omega)\Delta p_i^s + \omega\frac{1}{a_{ii}}(-(\frac{\tilde{\rho_i}}{\rho_0}-|\mathbf{F}_i^{adv}|)\tilde{\rho_i} - dterm(\Delta p^s)) \quad (13)$$

where ω is the relax coefficient.

4.2 Multiple-fluid Solver

For multiple-fluid simulations, the mixture model SPH solver solves the following equations [3]:

$$\frac{\partial \alpha_k}{\partial t} + (\mathbf{u}_m\cdot\nabla)\alpha_k = -\alpha_k\nabla\cdot\mathbf{u}_m - \nabla\cdot(\alpha_k\mathbf{u}_{mk}) \qquad (14)$$

$$\frac{\partial}{\partial t}\mathbf{u}_m + (\mathbf{u}_m\cdot\nabla)\mathbf{u}_m = -\frac{\nabla p_m}{\rho_m} + \frac{1}{\rho_m}\mathbf{M}_m \qquad (15)$$

where ρ_m is particle aggregate density, α_k is the volume fraction of the k-th phase in the particle, $\mathbf{u}_m, \mathbf{u}_{mk}$ are the particle velocity and k-th phase's drift velocity, $\mathbf{M}_m$ represents other sources of effective forces leading to particle velocity change. The phase velocity is $\mathbf{u}_k = \mathbf{u}_m + \mathbf{u}_{mk}$.

Since the particle composition is constantly changing in multiple-fluid flows, a J value for a fluid particle is not well-defined by default. Instead we use the phase-wise J_k and equation $\rho_m = \Sigma_k\alpha_k\rho_k$ to derive the incompressible solver. First, the density of mixture can be expressed as:

$$\rho_{m,t+1} = \Sigma_k\alpha_{k,t+1}\rho_k = \Sigma_k\frac{\alpha_{k,t}}{J_k}\rho_k \qquad (16)$$

where J_k is the Jacobian of phase-wise deformation gradient tensor. The last equality in Eqn.(16) is based on the assumption that each phase is uniformly "diluted" in the SPH particles. For the meta-volume within a particle, a rise in the volume fraction of phase k makes it denser, and vice versa, which is equivalent to local shrinking and expanding of phase-wise volumes. That is $J_k = \frac{\alpha_{k,t}}{\alpha_{k,t+1}}$.

Next, by solving the continuity equation in the mixture model framework [3] we can obtain $\alpha_{k,t+1}$ and treat $\rho_{m,t+1}$ on the left-hand side of Eqn.(16) as known values in the later computations. On the other hand, the intermediate $\mathbf{u}_m^{adv}$, $\mathbf{u}_k^{adv}$ and $\mathbf{x}_m^{adv}$ can be similarly calculated using the momentum equation, which provides an estimate of J_k from the mechanical-motion induced deformation. That is, we first compute $\mathbf{F}_k^{adv}$ with $\mathbf{u}_k$ using Eqn. (3), then $J_k^{adv} = |\mathbf{F}_k^{adv}|$. Ideally, one wishes to have $\Sigma_k\alpha_{k,t+1}\rho_k = \Sigma_k\frac{\tilde{\alpha_{k,t}\rho_k}}{J_k^{adv}}$ which is often not the case and needs a correction step. Here $\tilde{\alpha_{k,t}\rho_k}$ is interpolated value of $\alpha_{k,t}\rho_k$.

Similar to §4.1, a pressure correction field Δp_m is introduced, which will cause a velocity change of $\mathbf{u}_m^p$ on the SPH particle. Assuming the drift velocity does not change, the pressure correction leads to the velocity change of each phase $\mathbf{u}_k^p = \mathbf{u}_m^p$. The same derivation as Eqns. (4-10) applies. Using the first order Taylor expansion $(J_k+\Delta J_k)^{-1} \approx \frac{1}{J_k^2}(J_k - \Delta J_k)$, Eqn. (16) becomes

$$\rho_{mi,t+1} = \Sigma_k\frac{\tilde{\alpha_{ki,t}\rho_k}}{J_{ki}^{adv}} - \Sigma_k\frac{\tilde{\alpha_{ki,t}\rho_k}}{(J_{ki}^{adv})^2}\mathrm{tr}(\mathbf{F}_{ki}^{adv*}\mathrm{d}\mathbf{F}_{ki}^{adv}) \qquad (17)$$

The matrix-trace part in the above equation can be similarly developed by substituting $\mathbf{u}_k^p$ for $\mathbf{u}^p$ in Eqns. (6,9). Then we have

$$
\begin{aligned}
&\rho_{mi,t+1} - \Sigma_k\frac{\tilde{\alpha_{ki,t}\rho_k}}{J_{ki}^{adv}} \\
&= \Delta t\sum_{j\in \mathrm{NF(i)}} m_j(\mathbf{u}_{mij}^p)^T\underbrace{(\Sigma_k\frac{\tilde{\alpha_{ki,t}\rho_k}}{\tilde{\rho_{i,t}}(J_{ki}^{adv})^2}\mathbf{F}_{ki}^{adv*})^T}_{\mathbf{G}_i}\nabla W_{ij} \\
&= \Delta t\sum_{j\in \mathrm{NF(i)}} m_j(\mathbf{u}_{mij}^p)^T(\mathbf{G}_i)^T\nabla W_{ij} \qquad (18)
\end{aligned}
$$

Eqn. (18) has exactly the same form as Eqn. (10), and it follows the same derivation as well. The form of Eqn. (11) remains the same, only replacing $\rho_{\tilde{mi}}, \rho_{\tilde{mj}}$ for $\tilde{\rho_i}, \tilde{\rho_j}$. Eqn. (12) now becomes

$$
\begin{aligned}
\rho_{mi,t+1} - \Sigma_k\frac{\tilde{\alpha_{ki,t}\rho_k}}{J_{ki}^{adv}} &= (\sum_{j\in \mathrm{NF(i)}} m_j(\mathbf{d}_{ii}-\mathbf{d}_{ji})^T(\mathbf{G}_i)^T\nabla W_{ij})\Delta p_i \\
&+\sum_{j\in \mathrm{NF(i)}} m_j(\sum_{k\in NF(i)} \mathbf{d}_{ik}\Delta p_k - \mathbf{d}_{jj}\Delta p_j \\
&-\sum_{l\in \mathrm{NF(j)}-\{i\}} \mathbf{d}_{jl}\Delta p_l)^T(\mathbf{G}_i)^T\nabla W_i j \\
&= a_{ii}\Delta p_i + dterm(\Delta p) \qquad (19)
\end{aligned}
$$

The two differences in the derivation of Eqn. (19) from that of Eqn. (12) are the substitution of $\mathbf{G}_i$ for $\mathbf{F}_i^{adv*}$, and the replacement of the left-hand side. Solving the system using the relaxed Jacobi method completes the incompressible mixture-model framework. Eqn. (13) now becomes

$$\Delta p_i^{s+1} = (1-\omega)\Delta p_i^s + \omega\frac{1}{a_{ii}}(\rho_{mi,t+1} - \Sigma_k\frac{\tilde{\alpha_{ki,t}\rho_k}}{J_{ki}^{adv}} - dterm(\Delta p^s)) \qquad (20)$$

4.3 Boundary Handling

Similar to IISPH [6], we adopt the rigid fluid coupling technique [26] for boundary handling. A boundary particle b exerts a pressure force to multiple-fluid particle i:

$$\mathbf{\Gamma}_{i\leftarrow b}^p = -m_i\Psi(\rho_{mi})\frac{p_i}{\tilde{\rho_i}^2}\nabla W_{ib} \qquad (21)$$

which only substitute $\rho_{mi} = \Sigma_k\alpha_{ki}\rho_{ki}$ for ρ_{0i} in corresponding equations in Ψ compared to the single fluid form [6], [26]. The following derivations follow [6], and three modifications are applied. First,

$$
\begin{aligned}
\Delta t^2\frac{\mathbf{\Gamma}_i^p}{m_i} &= -\Delta t^2\sum_{j\in \mathrm{NF(i)}} m_j\left(\frac{p_i}{\tilde{\rho_i}^2}+\frac{p_j}{\tilde{\rho_j}^2}\right)\nabla W_{ij} - \Delta t^2\sum_{b\in \mathrm{NB(i)}}\Psi_b\frac{p_i}{\tilde{\rho_i}^2}\nabla W_{ib} \\
&= \underbrace{\left(-\Delta t^2\sum_{j\in \mathrm{NF(i)}}\frac{m_j}{\tilde{\rho_i}^2}\nabla W_{ij} - \Delta t^2\sum_{b\in \mathrm{NB(i)}}\frac{\Psi_b}{\tilde{\rho_i}^2}\nabla W_{ib}\right)}_{\mathbf{d}_{ii}} p_i \\
&+ \sum_{j\in \mathrm{NF(i)}}\underbrace{\left(-\Delta t^2\frac{m_j}{\tilde{\rho_j}^2}\nabla W_{ij}\right)}_{\mathbf{d}_{ij}} p_j \qquad (22)
\end{aligned}
$$

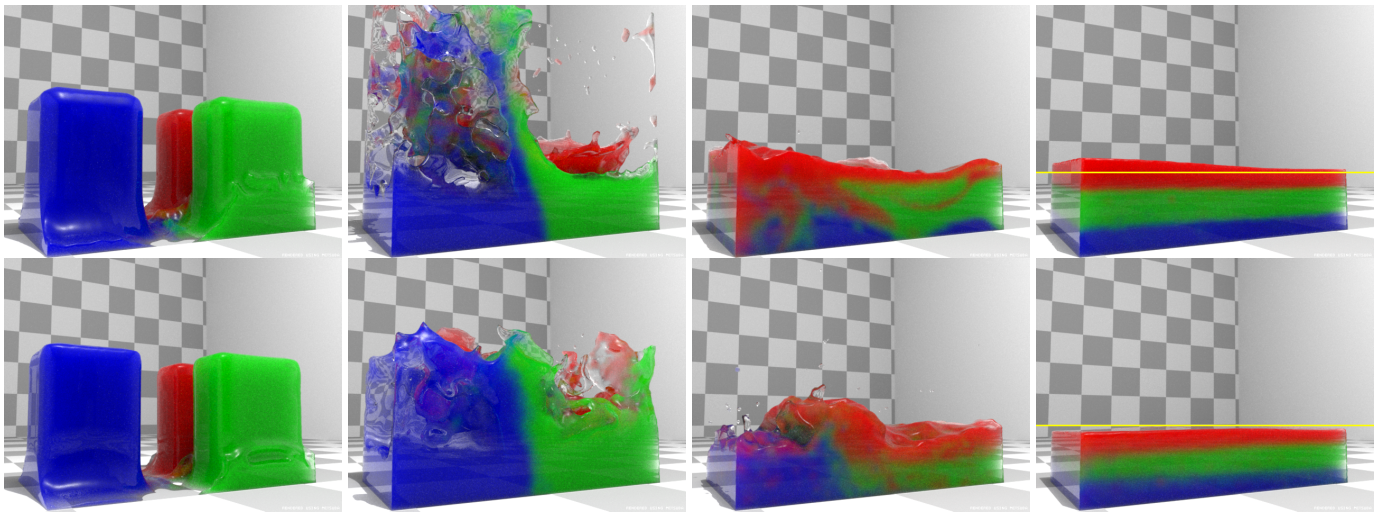


Fig. 1. Triple dambreak. Top row: result of our approach. Bottom row: result of [3]. Our approach significantly improves the incompressibility, better preserving fluid volume (indicated by the yellow line in the fourth column) and keeping ordered fine details (third column).

with an extra term $-\Delta t^2 \Sigma_{b\in NB(i)} \frac{\Psi_b}{\tilde{\rho}_i^2} \nabla W_{ib}$ added to $\mathbf{d}_{ii}$. Next, for single-fluid solver, substituting Eqn. (22) into Eqn. (10) yields

$$
\begin{aligned}
&-(\frac{\rho_i(t)}{\rho_0} - |F_i^{adv}|)\rho_i \\
&= \underbrace{(\sum_{j\in \mathrm{NF}(i)} m_j(\mathbf{d}_{ii} - \mathbf{d}_{ji})^T (\mathbf{F}_i^{adv*})^T \nabla W_{ij} + \sum_{b\in \mathrm{NB}(i)} \Psi_b \mathbf{d}_{ii}^T (\mathbf{F}_i^{adv*})^T \nabla W_{ib})\, p_i}_{a_{ii}} \\
&+ \underbrace{\sum_{j\in \mathrm{NF}(i)} m_j (\sum_{k\in \mathrm{NF}(i)} \mathbf{d}_{ik} p_k - \mathbf{d}_{jj} p_j - \sum_{l\in \mathrm{NF(j)}-\{i\}} \mathbf{d}_{jl} p_l)^T (\mathbf{F}_i^{adv*})^T \nabla W_{ij}}_{dterm(p)} \\
&+ \underbrace{\sum_{b\in \mathrm{NB}(i)} \Psi_b (\sum_{h\in \mathrm{NF}(i)} \mathbf{d}_{jh} p_h)^T (\mathbf{F}_i^{adv*})^T \nabla W_{ij}}_{bterm(p)} \\
&= a_{ii} p_i + dterm(p) + bterm(p)
\end{aligned} \tag{23}
$$

where an extra term $\Sigma_{b\in NB(i)} \Psi_b \mathbf{d}_{ii}^T (\mathbf{F}_i^{adv*})^T \nabla W_{ib}$ is added to a_{ii}, and an extra boundary term $bterm(\Delta p)$ is added to Eqn. (12). For the multiple-fluid solver, again by replacing the left hand side of Eqn. (23) with that of Eqn. (19) and using $\mathbf{G}_i$ in place of $\mathbf{F}_i^{adv*}$ in Eqn. (23), the corresponding linear system can be obtained.

4.4 Algorithm framework

The incompressible solver framework for mixture-model multiple fluid simulation is shown in Algorithm 1. The solver advances the simulation in a semi-implicit manner in that the drift velocity and the continuity equation are solved explicitly, and an implicit solver for pressure correction is applied to the momentum equation to enforce fluid incompressibility. This incompressible solver can be fully parallelized using GPU pipeline. Note that from the prediction step, our approach follows the same routine as the IISPH method, and can be readily integrated into the IISPH framework. Detailed discussions on solving such linear system with a relaxed Jacobi solver are provided in [6].

The stopping criteria are calculated as follows. For the single-fluid solver, we first subtract the right hand side from the left hand side of Eqn. (12) using Δp_i^{s+1} in the iteration, and then divide the result by ρ_0 to yield err_i:

$$
err_i \rho_0 = -(\frac{\tilde{\rho}_i}{\rho_0} - |\mathbf{F}_i^{adv}|)\tilde{\rho}_i - a_{ii}\Delta p_i^{s+1} - dterm(\Delta p_i^{s+1}) \tag{24}
$$

We show this err_i is equivalent to a relative density error. Using Eqn. (7), the relative error of predicted density can be expressed as $(\tilde{\rho}_i/J - \tilde{\rho}_i/J^*)/\rho_0$, where the superscript $*$ indicates the predicted value calculated using Δp_i^{s+1}. Assuming $J = 1 + \Delta J$, $J^* = 1 + \Delta J^*$ and using the first order Taylor expansion, the relative density error can be expressed as $(\tilde{\rho}_i/\rho_0)(J^* - J) = [-\tilde{\rho}_i(J - J^*)]/\rho_0$, which is just err_i by comparing to Eqn. (7) and Eqn. (12). For the multiple-fluid solver, similarly, we first subtract the right hand side from the left hand side of Eqn. (19) using Δp_i^{s+1} in the iteration, and then divide the result by $\rho_{mi,t+1}$ to yield err_i:

$$
err_i \rho_{mi,t+1} = \rho_{mi,t+1} - \Sigma_k \frac{\alpha_{ki,\tilde{t}} \rho_k}{J_{ki}^{adv}} - a_{ii}\Delta p_i^{s+1} - dterm(\Delta p_i^{s+1}) \tag{25}
$$

This err_i is also equivalent to a relative density error. In the multiple-fluid solver, a predicted mixture density is computed by the right hand side of Eqn. (17) using the Jacobian of deformation gradients, and err_i is its relative error.

The error parameter err is calculated as the maximum err_i in both single-fluid or multiple-fluid cases. We then compare err to a constant $\eta = 0.1\%$ at the start of each iteration in the relaxed Jacobi solver. We also set a maximum iteration number $maxIter = 20$. The relaxed Jacobi solver stops the iteration when $err \leq \eta$ or $maxIter$ is reached. Same with the IISPH method, negative pressure values are clamped to zero in the solver.

The choice of stopping criteria is analyzed in Fig. 3 using the triple dambreak case in Fig. 1. With $\eta = 0.1\%$, the convergence performance of our solver is tested under $maxIter = 20, 50, 100$, respectively. We show both the "predict" average density error evaluated when the relaxed Jacobi solver ends iteration (left) and the "actual" average density error evaluated at the next time step using actual

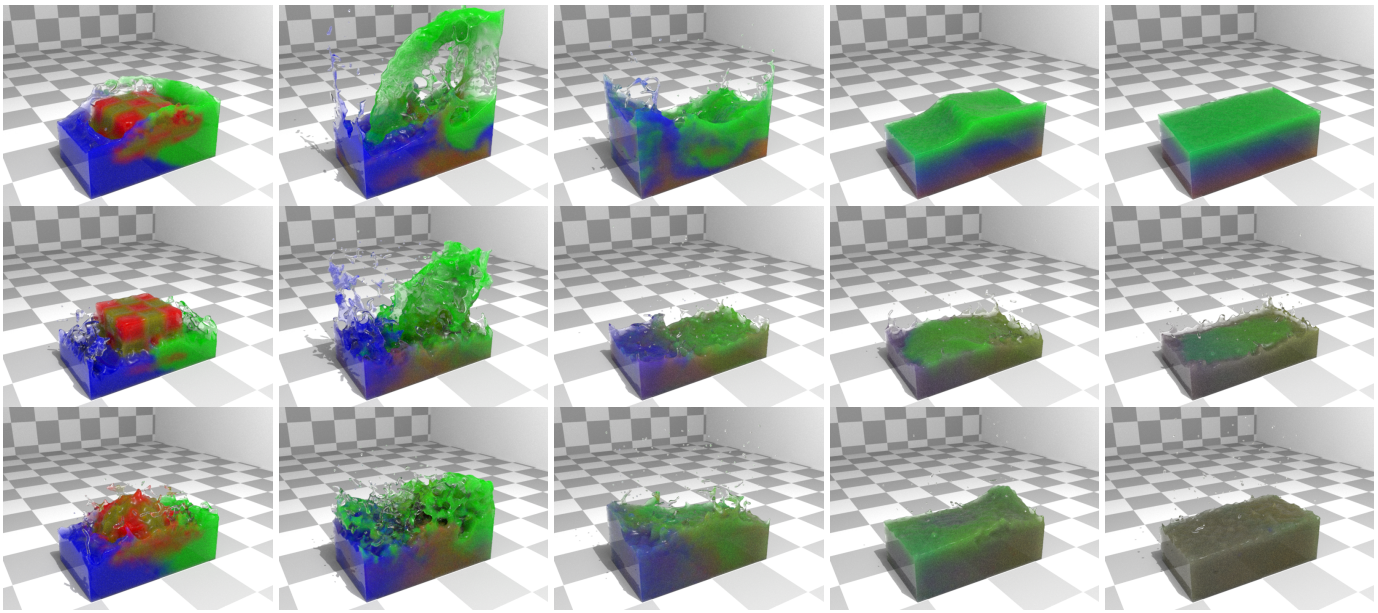


Fig. 2. Cube fall. Top: a cube falls into the underlying pool in a four-phase miscible setting. Our approach successfully recovers incompressible flow motion. Middle: using the method of [3]. Violent boundary oscillation and unnatural mixing color results from high compressibility. Bottom: using the method of [3] with $1/100$ time step size and $100\times$ stiffness. Total volume is reasonably preserved but the color-mixing and boundary issues remain.

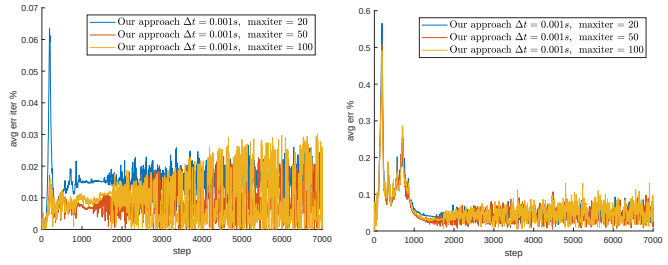


Fig. 3. Average density error using different $maxIter$ value. Left: "predict" average density error evaluated when the relaxed Jacobi solver ends iteration. Right: "actual" average density error evaluated at the next time step using actual particle neighborhood.

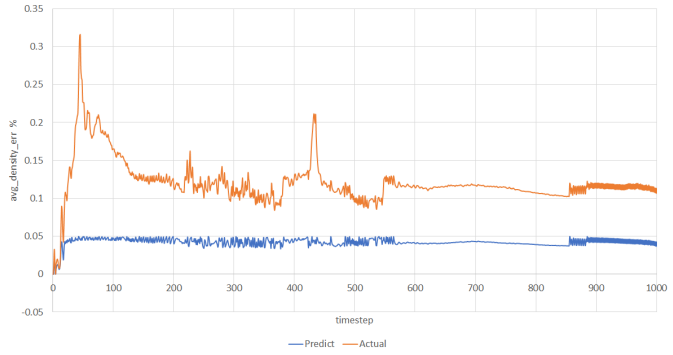


Fig. 4. The "predict" and "actual" average density errors in a single fluid IISPH simulation using the method in [6]. Larger "actual" errors at the next time step with a similar double-peak-shape curve can be observed.

particle neighborhood (right), i.e. $(\tilde{\rho}_i - \Sigma_k \alpha_{ki}\rho_k)/\Sigma_k \alpha_{ki}\rho_k$, which is the relative error of interpolated density from theoretical mixture density. As shown on the left of Fig. 3, the average density error within the relaxed Jacobi solver is less than 0.1% using $maxIter = 20$ and converges to about 0.02% during the simulation. Increasing $maxIter$ to 50 and 100 leads to a generally smaller average density error within the relaxed Jacobi solver, but with larger variance between time steps. On the right of Fig. 3, we also calculate the average density error at the beginning of the next time step using the actual particle neighborhood. Larger errors occur in the simulator, but the largest error is still below 0.6% and in general below 0.2%, which converges to around 0.05% when the flow becomes more steady. It is noted that the larger "actual" error at the next time step also shows up in the original IISPH framework. Fig. 4 shows the "predict" and "actual" average density errors of a single-fluid IISPH simulation using the method in [6]. Larger errors at the next time step are also observed. Interestingly, although increasing $maxIter$ value to above 20 leads to improvements on the "predict" average density errors, it has only small benefits on the "actual" average density errors. In cases of $maxIter = 50$ and $maxIter = 100$, under maximum-error criterion, the average converged iteration numbers are 23.8 and 22.1, respectively, and the minimum iteration needed is about 10 in all cases. On the other hand, Fig. 3 shows $maxIter = 20$ has achieved satisfactory actual average errors during the simulation. As a result we use $maxIter = 20$ as a balanced choice.

In the prediction step, previous methods [6], [20] left out the pressure force in predicting intermediate states. However in our experiments we find that for multiple-fluid simulation it is better to include explicitly-calculated pressure force in the prediction step. One possible reason is that better initial value is provided this way.

5 Results

In this section, several comparison examples are presented to examine the effectiveness of the proposed incompressible

Algorithm 1 Incompressible Mixture-model Solver

for all particles **do**
  compute drift-velocity u_{mk} as in [3]
end for
for all particles **do**
  solve the continuity equation and obtain $\alpha_{k,t+1}, \rho_{m,t+1}$
end for
for all particles **do**
  predict $\mathbf{x}_m^{adv}, \mathbf{u}_m^{adv}, \mathbf{u}_k^{adv}, \mathbf{F}_k^{adv}, J_k^{adv}$ with momentum equation
end for
$s = 0$
while $err > \eta \&\& s < maxIter$ **do**
  for all particle i **do**
    calculate $\Sigma_j \mathbf{d}_{ij} \Delta p_j^s$ in Eqn.(11)
  end for
  for all particles **do**
    compute Δp_i^{s+1} with Eqn.(13) or Eqn.(20)
  end for
  $s = s + 1$
end while
for all particles **do**
  compute pressure correction force
  correct particle position and velocity
end for

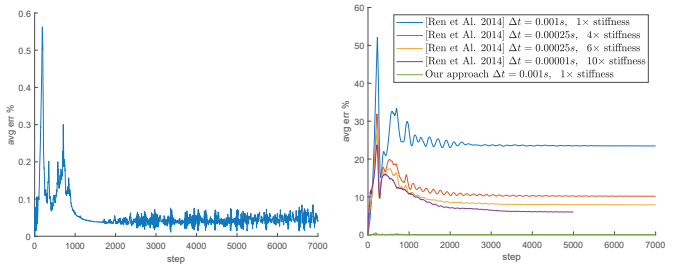


Fig. 5. Average density error in the triple dambreak case. Left: our approach. Right: comparison of the previous approach [3] in different timesteps and stiffness values with our approach. Our approach achieves significantly better incompressibility. The "$1\times$ stiffness" of our approach is only used in the prediction step as described in §4.4.

SPH solver, and we also demonstrate how the new solver can be further exploited to provide artistic control. The new algorithm is implemented on a Nvidia GTX 1080Ti GPU, for which the performance data of all examples are listed in Table 1. The scalability is affected by the total particle number and the phase number, both of which linearly raise the related computational cost. Generally, our algorithm shows sub-linear scalability relative to "particle number times phase number" value. In the comparisons we control the time step sizes to be the same with previous methods ($0.001s$ for both multiple-fluid [3] and single-fluid [6]). The readers are referred to the supplemental video which shows closer visual observation and clearer comparisons.

Fig. 1 shows a comparison between our incompressible solver and the original mixture model solver [3], using an immiscible triple dambreak scene (density ratio red:green:blue = 1:2:3). The average density error during the simulation is plotted in Fig. 5, which shows the "actual" error discussed in §4.4. Our incompressible solver significantly reduces the density error and produces much better visual effects in the result. Note that the fluid volume is better maintained in the last column, and fine details have more ordered appearance in the third column.

Fig. 2 shows a cube falling into the pool in a four-phase miscible setting, with the density ratio green:blue:red:yellow = 1:2:3:3. Our approach performs well with incompressible flow motion. The same scene is simulated using the previous method [3] as shown in Fig. 2 middle row. Due to the high non-physical compressibility of the previous method, excessive particle penetration occurs at the particle boundary, and an extra soft boundary is required to hold them back. Still, large compressibility and violent oscillation near the boundary make the mixing effect of [3] unnatural. It is worth noting that for this example the previous method must run at a relatively low stiffness value using the same time step size. Larger stiffness value will make its boundary issue more severe and even cause break-down to the simulation. This leads to a severe volume loss in the middle row of Fig. 2 under an ordered initial particle position setting due to failure of volume holding at the beginning of simulation. In our experiment, the middle row has a volume loss up to 48.7% and converges to 41.2%. The result of [3] run at $1/100$ time step and $100\times$ stiffness is shown in the bottom row of Fig. 2, the volume loss is reduced to 21.9% in the beginning and 9.1% in the end. However, artifacts still remain and the excessive compressive oscillation still thoroughly mix the components together resulting in a dull visual appearance. Our new approach provides natural looking results without artifacts.

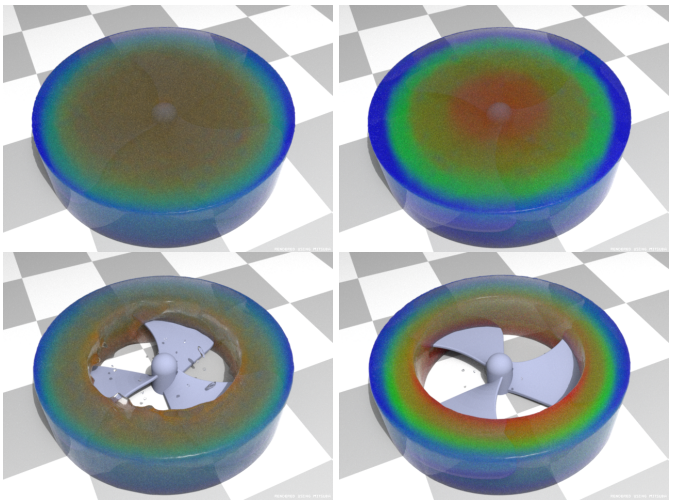


Fig. 6. Centrifugal separation. Top row: result of our approach. Bottom row: result of [3]. Our incompressible solver successfully avoids the central void due to compressibility of the previous method.

In Fig. 6, an un-mixing scene with large centrifugal force is tested. The cylindrical container is totally sealed and filled up with fluid, and there should be no free surface at all if ideal incompressibility is enforced. In the initial state, four immiscible phases (density ratio red:yellow:green:blue = 1:1.5:2.5:3) are artificially mixed together with the volume fraction ratio red:yellow:green:blue = 0.15:0.2:0.25:0.4. Our incompressible approach successfully avoids the large cen-

TABLE 1
Performance

Example Name	Phase Number	Liquid Particle Number	runtime-our (second/step)	runtime-previous [3] (second/step)
Dambreak	3	62,000	0.148	0.037
Cube Fall	4	89,000	0.197	0.071
Unmixing	4	300,000	0.282	0.177
Control	2	131,000	0.151	-

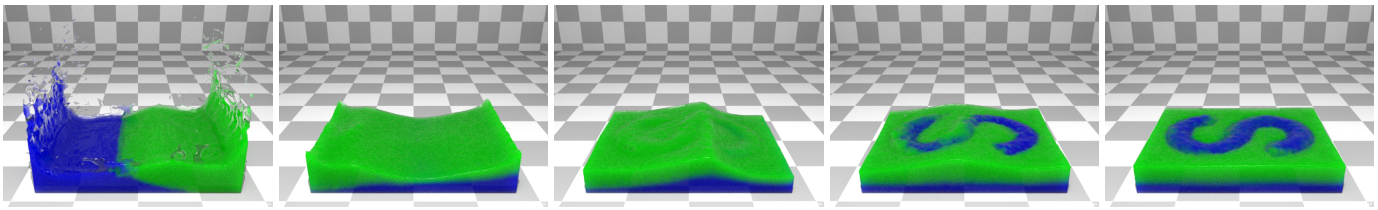


Fig. 7. Artistic control. In the middle of the simulation, we artificially raise the target Jacobian of the SPH particles in a S-shaped region. The heavier blue phase forms a fountain in the region but the other areas remains in steady layered flow appearance.

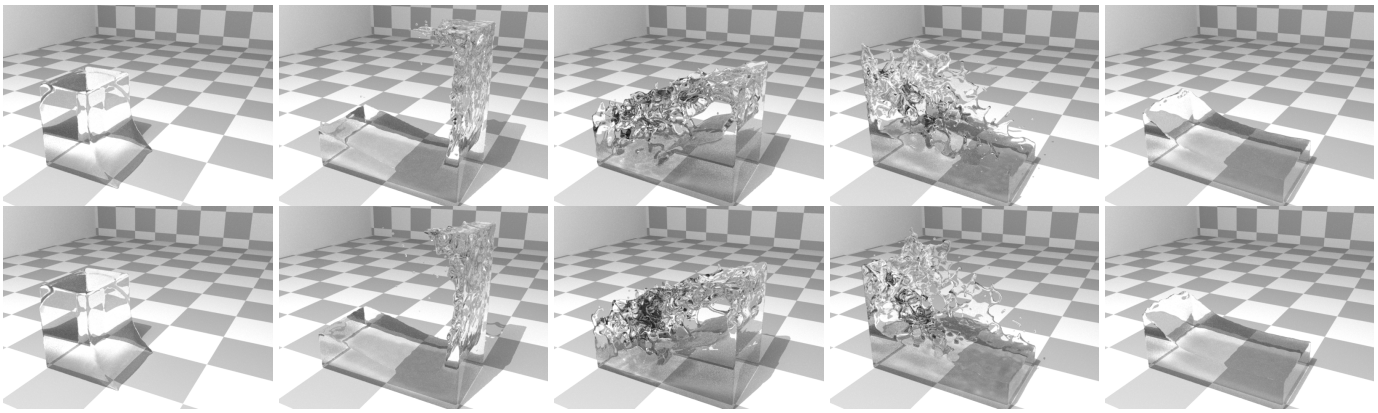


Fig. 8. Single-fluid simulation test. Top row: result of our approach. Bottom row: result of IISPH [6]. Our approach achieves comparable incompressible visual effects with IISPH.

tral void which always appears in the result of previous approach [3].

In Fig. 7, the deformation gradient is utilized to control artistic effects. Specifically, the effective particle volume can be controlled using a user-specified J without changing the phase density. Since $J_{m,t} = \frac{V_{m,t+1}}{V_{m,t}}$, a controlling volume-expansion coefficient J_{cont} can be introduced so that the particle volume in the next time step can be set to any desired value $\bar{V}_{m,t+1} = J_{cont} J_{m,t} V_{m,t}$. Rewriting Eqn. (16) as $\frac{\rho_{m,t}}{J_{m,t}} = \Sigma_k \frac{\alpha_{k,t}}{J_k} \rho_k$, the formulation in §4.2 effectively finds a pressure correction that satisfies the target $J_{m,t}$. To change the target $J_{m,t}$ to $J_{cont} J_{m,t}$, one only needs to substitute $\frac{\rho_{mi,t+1}}{J_{cont}}$ for $\rho_{mi,t+1}$ in Eqn. (18). This controlling scheme has an interesting effect that the particles with controlled J have a larger pressure if $J_{cont} < 1$ in the simulation that can affect our prediction step, and vice versa. As a result, after applying particle volume control $J_{cont} = 2$ in the middle of the two-phase scene as shown in Fig. 7 (with the density ratio green:blue = 1:2), the heavier blue phase formed a fountain within our user-defined S-shape control area, but the other areas remains in steady layered flow appearance.

We show that using deformation gradient is equivalent to using constant-density in single-fluid simulations. In Fig. 8, the performance of our approach is tested on a single-fluid simulation. All parameters are set at the same values in the comparison between our algorithm and the IISPH algorithm. Our approach achieves comparable incompressibility in the single fluid simulation. To quantitatively evaluate the performance and convergence of our incompressible solver, we re-run this dambreak scene with a fixed iteration number both for IISPH and our approach. The fixed iteration number is set to 10 which is the average iteration number of IISPH before convergence (which is after 9-11 iterations) in Fig. 8. In this fixed-iteration-number test using 52,000 liquid particles, our approach runs at $1.73s$ per step and IISPH runs at $1.53s$ per step using a single core on an Intel Xeon 2.60GHz CPU. The average density error and the max density error during the simulation are shown in Fig. 9, reflecting almost the same converging speed.

To demonstrate the effectiveness of our method, we also examine how good the result of [3] can achieve at similar or more computational costs compared with ours. As Table 1 shows our method costs 2-4 times as much, we first run the method of [3] using $1/4$ time step size (0.00025s) with higher stiffness values. The diagram on the right side of Fig. 5 shows a quantitative result of the average density using

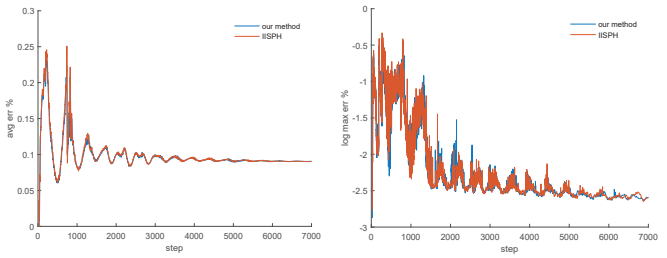


Fig. 9. Quantitative evaluation in single-fluid simulation. Left: average density error. Right: max density error. Our approach achieves comparable results with IISPH in the single-fluid simulation.

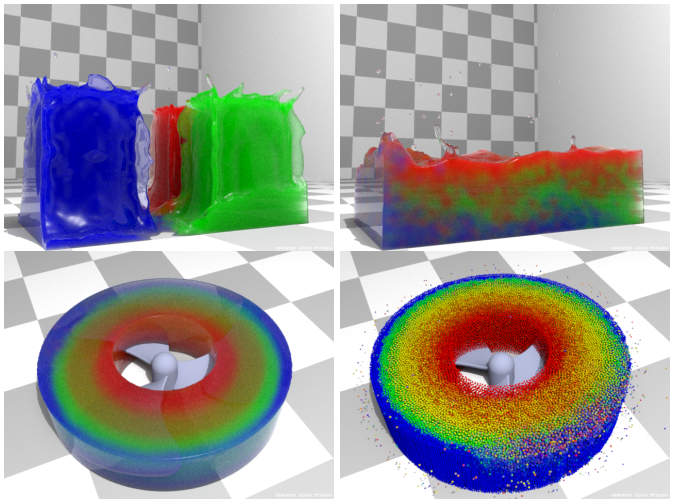


Fig. 10. Results of tuning up stiffness and reducing time step using [3], they still do not reach the quality of our approach. Top row: triple dambreak case with $1/100$ time step size and $10\times$ stiffness, render timings are the same with the first and third columns in Fig.1. Bottom left: centrifugal separating case with $1/4$ time step and $14\times$ stiffness. Bottom right: centrifugal separating case with $1/4$ time step and $16\times$ stiffness, particles start to fly out of the scene while the central void still being obvious.

[3] under different time step sizes and stiffness values. These errors never come close to 5%, but our method achieves less-than 0.5% error during the simulation. We further reduce the time step size to $1/100$, which allows larger stiffness to be stable. However our experiments show that simply tuning up the stiffness has limited effects on ensuring incompressibility. For the dambreak case, as can be seen in Fig. 5 and Fig. 10 top row, the shape of the liquid block has already started to collapse at $10\times$ stiffness even at such small time step and still with relatively large density error. A similar visual shape collapsing under high stiffness value can also be observed in the Cube Fall case in the bottom row of Fig. 2. The bottom left of Fig. 10 shows the centrifugal separating case at $1/4$ time step and $14\times$ stiffness, any larger stiffness value will cause instability of simulation (bottom right), but the central void still remains. In all these cases, our approach can more efficiently produce incompressible mixture-model effects with less parameter tuning efforts.

In Fig. 11, we render the triple dambreak case in Fig. 1 in particle view. On the right hand side we can observe large color variances between adjacent particles, resulting in

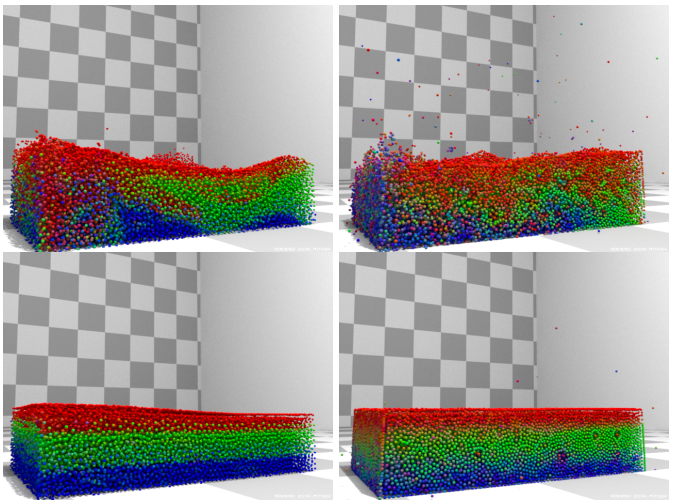


Fig. 11. Particle view of the triple dambreak case. Left: our approach. Right: Previous approach [3]. Our approach shows more ordered particle layering and better particle distributions.

a dispersed visual appearance. There are also irregular particle distribution patterns in bottom right on the boundary. Our result on the left hand side shows more ordered particle color layering and improves the particle distribution in the simulation.

Acknowledgments

This work is supported by the National Key R&D Program of China(2017YFB1002701). We would also like to thank all the anonymous reviewers for the inspiring suggestions.

6 Conclusion

Based on the deformation gradient, we derive a novel incompressibility enforcement scheme for SPH fluid simulations. The proposed approach decouples the non-physical compressibility from the fluid density and the divergence of velocity in the theoretic formulation. It greatly enhances the visual effect of multiple-fluid simulations using the mixture model, significantly reducing the compressibility in the previous solver [3]. The proposed algorithm is fully parallelizable, and can be readily implemented into previous IISPH framework.

The proposed scheme does not fully take advantage of single-fluid properties such as the divergence-free condition. Although the DFSPH method [20] does not apply on multiple fluid simulations for it requires divergence free, on single-fluid simulations, our approach has larger divergence among particles than the DFSPH method. This is a limitation of our approach for single-fluid simulations where strict enforcement of divergence free property is usually desired. In multiple-fluid simulations, occasionally a small amount of particles can take a long time to converge to a given error threshold and we bound the max iteration in the implementation. One possible cause is that the proposed approach uses a semi-implicit solver for the mixture-model SPH simulation, which uses an explicit scheme for the continuity equation. Deformation gradients do not evolve in

an additive manner. Another possible cause is that we use the maximum error as criterion and it is harder to satisfy than an average-error criterion, which may also lead to the large variance in the predict density error. In practice, we evaluate the function between two adjacent time steps to obtain a linear local approximation in Eqn.(3), which is found to give adequate results. However, higher-order approximations are interesting topics and can be studied in the future work. Moreover, reducing the gap between predicted and actual errors is worth future study. Recently strong fluid-rigid coupling techniques such as [27] have been proposed achieving impressive results. Integrating them into our incompressible multiple fluid framework worths investigation, for they may further enhance the stability and convergence at solid boundary. Some researches use particle labeling to simulate immiscible multiple fluids within single fluid simulation frameworks. Applying our approach on those methods will require adjustment of the deformation gradient calculations near phase interfaces and we would also like to investigate it in the future. Another future direction is providing a fully implicit solver which ensures exact mass continuity in the calculation. We would also like to investigate the application of our approach in solid-related phenomena such as dissolving.

REFERENCES

[1] T. Yang, J. Chang, B. Ren, M. C. Lin, J. J. Zhang, and S.-M. Hu, "Fast multiple-fluid simulation using helmholtz free energy," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 201:1–201:11, Oct. 2015. [Online]. Available: http://doi.acm.org/10.1145/2816795.2818117

[2] T. Yang, J. Chang, M. C. Lin, R. R. Martin, J. J. Zhang, and S.-M. Hu, "A unified particle system framework for multi-phase, multi-material visual simulations," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 224:1–224:13, Nov. 2017. [Online]. Available: http://doi.acm.org/10.1145/3130800.3130882

[3] B. Ren, C. Li, X. Yan, M. C. Lin, J. Bonet, and S.-M. Hu, "Multiple-fluid sph simulation using a mixture model," *ACM Trans. Graph.*, vol. 33, no. 5, pp. 171:1–171:11, Sep. 2014. [Online]. Available: http://doi.acm.org/10.1145/2645703

[4] X. Yan, Y.-T. Jiang, C.-F. Li, R. R. Martin, and S.-M. Hu, "Multiphase sph simulation for interactive fluids and solids," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 79:1–79:11, Jul. 2016. [Online]. Available: http://doi.acm.org/10.1145/2897824.2925897

[5] J. Aboudi, S. Arnold, and B. Bednarcyk, *Micromechanics of Composite Materials: A Generalized Multiscale Analysis Approach*, 01 2013.

[6] M. Ihmsen, J. Cornelis, B. Solenthaler, C. Horvath, and M. Teschner, "Implicit incompressible sph," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 3, pp. 426–435, Mar. 2014. [Online]. Available: http://dx.doi.org/10.1109/TVCG.2013.105

[7] J. Monaghan, "Simulating free surface flows with sph," *Journal of Computational Physics*, vol. 110, no. 2, pp. 399 – 406, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999184710345

[8] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '03. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003, pp. 154–159. [Online]. Available: http://dl.acm.org/citation.cfm?id=846276.846298

[9] M. Becker and M. Teschner, "Weakly compressible sph for free surface flows," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '07. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 209–217. [Online]. Available: http://dl.acm.org/citation.cfm?id=1272690.1272719

[10] S. J. Cummins and M. Rudman, "An sph projection method," *Journal of Computational Physics*, vol. 152, no. 2, pp. 584 – 607, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999199962460

[11] S. Premoze, T. Tasdizen, J. Bigler, A. Lefohn, and R. T Whitaker, "Particle-based simulation of fluids," *Computer Graphics Forum*, vol. 22, 06 2003.

[12] F. Losasso, J. Talton, N. Kwatra, and R. Fedkiw, "Two-way coupled sph and particle level set fluid simulation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 4, pp. 797–804, Jul. 2008. [Online]. Available: http://dx.doi.org/10.1109/TVCG.2008.37

[13] Y. Gao, C.-F. Li, S. Hu, and B. A. Barsky, "Simulating gaseous fluids with low and high speeds," *Comput. Graph. Forum*, vol. 28, pp. 1845–1852, 2009.

[14] D. A. Fulk and D. W. Quinn, "Hybrid formulations of smoothed particle hydrodynamics," *International Journal of Impact Engineering*, vol. 17, no. 1, pp. 329 – 340, 1995, hypervelocity Impact. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0734743X9599859P

[15] T. Harada, S. Koshizuka, and Y. Kawaguchi, "Smoothed particle hydrodynamics on gpus," *Computer Graphics International*, 01 2007.

[16] B. Solenthaler and R. Pajarola, "Predictive-corrective incompressible sph," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 40:1–40:6, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1531326.1531346

[17] K. Bodin, C. Lacoursière, and M. Servin, "Constraint fluids," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, pp. 516–526, 2012.

[18] N. Kang and D. Sagong, "Incompressible sph using the divergence-free condition," *Computer Graphics Forum*, vol. 33, 10 2014.

[19] M. Macklin and M. Müller, "Position based fluids," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 104:1–104:12, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2461912.2461984

[20] J. Bender and D. Koschier, "Divergence-free sph for incompressible and viscous fluids," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, pp. 1–1, 06 2016.

[21] X. Yan, C.-F. Li, X.-S. Chen, and S.-M. Hu, "Mpm simulation of interacting fluids and solids," *Computer Graphics Forum*, vol. 37, pp. 183–193, 12 2018.

[22] A. Pradhana Tampubolon, T. Gast, G. Klar, C. Fu, J. Teran, C. Jiang, and K. Museth, "Multi-species simulation of porous sand and water mixtures," *ACM Transactions on Graphics*, vol. 36, pp. 1–11, 07 2017.

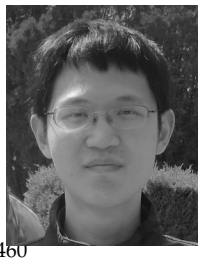
[23] M. Gao, A. Pradhana, X. Han, Q. Guo, G. Kot, E. Sifakis, and C. Jiang, "Animating fluid sediment mixture in particle-laden flows," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 149:1–149:11, Jul. 2018. [Online]. Available: http://doi.acm.org/10.1145/3197517.3201309

[24] C. H. Lee, A. J. Gil, O. I. Hassan, J. Bonet, and S. Kulasegaram, "A variationally consistent streamline upwind petrov-galerkin smooth particle hydrodynamics algorithm for large strain solid dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 318, pp. 514 – 536, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0045782516315201

[25] E. A. de Souza Neto, D. Peric, and D. R. Owen, *Computational methods for plasticity: theory and applications*. John Wiley & Sons, 2011.

[26] 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, pp. 62:1–62:8, Jul. 2012. [Online]. Available: http://doi.acm.org/10.1145/2185520.2185558

[27] 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, Jan. 2019. [Online]. Available: https://doi.org/10.1145/3284980



Bo Ren received the PhD degree from Tsinghua University in 2015. He is currently an associate professor in the College of Computer Science, Nankai University, Tianjin. His research interests include physically-based simulation and rendering, scene geometry reconstruction and analysis.

Wei He received the B.S. degree in intelligent science and technology from Nankai University, Tianjin, China, in 2019, where he is currently working toward the M.S. degree in control science and engineering at the Institute of Robotics and Automatic Information Systems. His research interests include motion planning, state estimation and intelligent control of robots, especially unmanned aerial vehicles and aerial manipulation systems.

Chen-feng Li is currently a Professor at the College of Engineering of Swansea University, UK. He received his BEng and MSc degrees from Tsinghua University, Beijing in 1999 and 2002, respectively, and received his Ph.D. degree from Swansea University in 2006. His research interests include computational solid mechanics, computational fluid dynamics, and computational graphics. He is the Editor-in-Chief of Engineering Computations.

Xu Chen is expected to receive his bachelor degree in software engineering from Nankai University (Tianjin, China) in 2020. His research interests lie in computer graphics with particular on physically-based simulation and data science.