Kinetic Simulation of Turbulent Multifluid Flows

Wei Li, Kui Wu, Mathieu Desbrun

Kinetic Simulation of Turbulent Multifluid Flows

WEI LI, LightSpeed Studios, China
KUI WU, LightSpeed Studios, USA
MATHIEU DESBRUN, Inria / Ecole Polytechnique, France

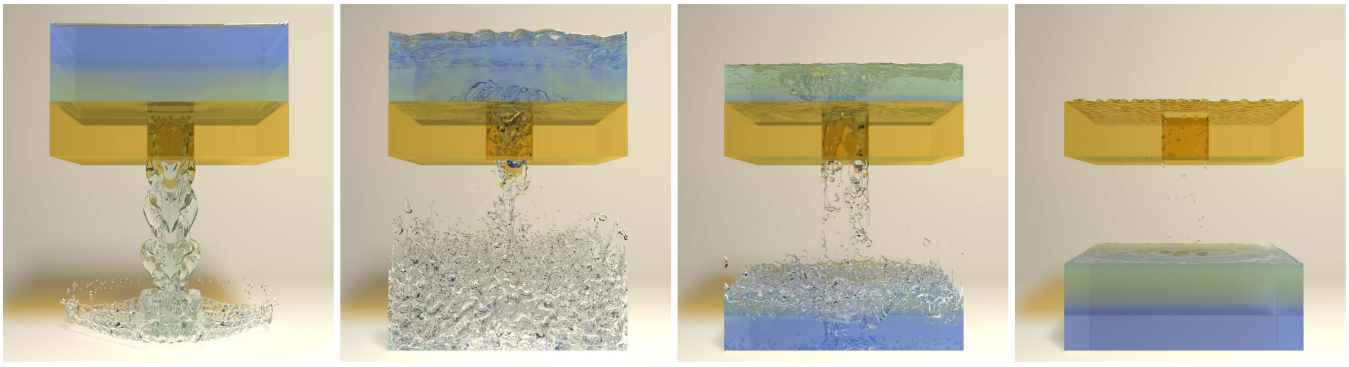


Fig. 1. **Glugging with Rayleigh–Taylor instability:** Our multiphase solver produces guggling for two colored immiscible fluids in the transparent upper container and air in the bottom one (water:oil:air with density ratios 800 : 400 : 1), with a square conduit in between. The two fluids are initially arranged with the heavier fluid (blue) on top of the lighter fluid (cyan), thus forming Rayleigh–Taylor instabilities early on, before flowing down and exhibiting splashes, bubbles and wetting, and ending up in stable layers based on their densities.

Despite its visual appeal, the simulation of separated multiphase flows (i.e., streams of fluids separated by interfaces) faces numerous challenges in accurately reproducing complex behaviors such as guggling, wetting, or bubbling. These difficulties are especially pronounced for high Reynolds numbers and large density variations between fluids, most likely explaining why they have received comparatively little attention in Computer Graphics compared to single- or two-phase flows. In this paper, we present a full LBM solver for multifluid simulation. We derive a conservative phase field model with which the spatial presence of each fluid or phase is encoded to allow for the simulation of miscible, immiscible and even partially-miscible fluids, while the temporal evolution of the phases is performed using a D3Q7 lattice-Boltzmann discretization. The velocity field, handled through the recent high-order moment-encoded LBM (HOME-LBM) framework to minimize its memory footprint, is simulated via a velocity-based distribution stored on a D3Q27 or D3Q19 discretization to offer accuracy and stability to large density ratios even in turbulent scenarios, while coupling with the phases through pressure, viscosity, and interfacial forces is achieved by leveraging the diffuse encoding of interfaces. The resulting solver addresses a number of limitations of kinetic methods in both computational fluid dynamics and computer graphics: it offers a fast, accurate, and low-memory fluid solver enabling efficient turbulent multiphase simulations free of the typical oscillatory pressure behavior near boundaries. We present several numerical benchmarks, examples and comparisons of multiphase flows to demonstrate our solver's visual complexity, accuracy, and realism.

Authors' addresses: Wei Li (eduardli@tencent.com): LightSpeed Studios, Shanghai, China; Kui Wu (kwwu@global.tencent.com): LightSpeed Studios, Los Angeles, USA; Mathieu Desbrun (mathieu.desbrun@inria.fr): Inria Saclay/LIX, Institut Polytechnique de Paris, Palaiseau, France.

CCS Concepts: • **Computing methodologies → Physical simulation**.

Additional Key Words and Phrases: Lattice Boltzmann method, velocity moments, turbulent flows, large density ratio, multifluid flows.

ACM Reference Format:
Wei Li, Kui Wu, and Mathieu Desbrun. 2024. Kinetic Simulation of Turbulent Multifluid Flows. *ACM Trans. Graph.* 43, 4, Article 55 (July 2024), 17 pages. https://doi.org/10.1145/3658178

1 INTRODUCTION

Fluid simulation in graphics has been studied for decades now, as the visual intricacy of fluid motions remains highly sought after. While airflow simulation has reached an impressive level of realism, real-world flows often involve multiple fluids (of various densities, viscosities, etc.) and air. This type of "multiphase flows" presents challenges for both Computational Fluid Dynamics (CFD) and Computer Graphics (CG), since modeling the equations of motion for mixtures of oil, gas, and water in engineering or realistically simulating bubbling and splashing for multiple fluids in visual effects remains difficult. In particular, the rare works dedicated to multiphase flow simulation in CG all come with strong limitations: current solvers often cannot efficiently handle fluids with large density ratios [Losasso et al. 2006; Mihalef et al. 2007; Kim 2010; Premžoe et al. 2003; Müller et al. 2005; Ren et al. 2014], introduce too much numerical dissipation to handle turbulence at high Reynolds numbers [Yan et al. 2018; Yan and Ren 2023], or only apply to two-phase flows [Hong and Kim 2005; Ando et al. 2015; Li et al. 2021, 2022; Li and Desbrun 2023].

In this work, we introduce a multifluid flow simulation approach offering a unified n-phase treatment of a large variety of (im)miscible multiphase fluid phenomena, i.e., bubbling, wetting, splashing, glugging, and mixing. Our approach borrows from recent advances in both phase and velocity motions in CFD and CG. In particular, we

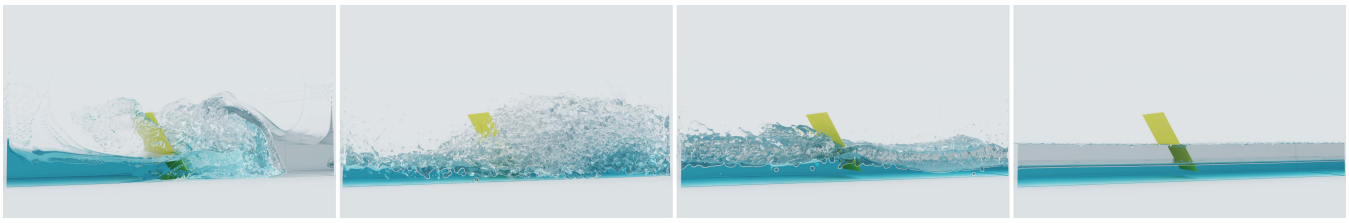


Fig. 2. **Dam breaking over thin shell.** A heavier blue liquid hits a thin obstacle first and forms an obvious crown splash, then rushes into the lighter gray liquids, forming a large amount of turbulence and bubbles. The two fluids end up separated at the end.

offer a low-memory footprint for both the n phases (through a simple D3Q7 LBM discretization) and the velocity field (using D3Q27 HOME-LBM, which stores only the first three velocity moments instead of the associated 27 distribution values for each grid node of the simulation grid). We also extend recent two-fluid solvers so that n fluids, encoded by n constrained (im)miscible phases, can be easily simulated based on their respective density and viscosity. The resulting solver is compared to previous works and evaluated across various scenarios in order to highlight its efficiency, its memory consumption, as well as its realism.

2 RELATED WORK

Before delving into our contributions, we provide some context by briefly reviewing related work on multiphase fluid simulation from computer graphics (CG) and computational fluid dynamics (CFD).

2.1 N-phase fluid solvers in CG

CG approaches for the simulation of multiple phases (or multiple fluids) fall roughly into three categories: grid-based, particle-based, or hybrid methods, based on the discretization of the fluids.

Grid-based methods. Building upon Stable Fluids [Stam 1999], Hong and Kim [2005] added jump conditions in the pressure field around the free surface, while Nielsen and Østerby [2013] used a two-continua Poisson equation to induce spray, marking the beginning of Eulerian multiphase simulation in CG. The addition of volume-of-fluid (VOF) or level-set interface discretization allowed for sharp or smooth transitions between fluids [Bao et al. 2010; Kang et al. 2010]. Leveraging lattice Boltzmann methods (LBM) to simulate two miscible fluids was first proposed in [Zhu et al. 2006, 2007]. Recent two-phase kinetic solvers have dramatically improved efficiency, stability to high Reynolds numbers, and versatility as they support bubbling, glugging, wetting, splashing, two-way fluid-solid coupling, and high-density ratios [Li et al. 2021, 2022; Li and Desbrun 2023].

Particle-based methods. Particle-based methods are Lagrangian by nature, making particle/solid interaction far simpler than Eulerian methods. In this category, Premžoe et al. [2003] first proposed the Moving Particle Semi-implicit (MPS) method to simulate immiscible flows, as it is easy to track the motion of individual particles with its own intrinsic properties (density, viscosity, etc). Soon after, Müller et al. [2005] contributed a Smoothed Particle Hydrodynamics (SPH) method based on [Desbrun and Gascuel 1996] to handle interactions between different fluids, including both immiscible and miscible liquids. SPH-based multiphase solvers were further extended to handle chemical reactions [Ren et al. 2014], partial dissolution [Yang et al. 2015], solid-liquid coupling [Yan et al. 2016; Yang et al. 2017], porous materials handling [Yang et al. 2017], incompressibility [Ren et al. 2021; Jiang et al. 2020], and high relative phase motions [Jiang and Lan 2021]. Most recently, Yan and Ren [2023] proposed the use of peridynamic mixture-model theory to handle high-density ratios between fluids, rivaling in that aspect with LBM-based two-fluid solvers — but no turbulence or bubbles were generated as Lagrangian approaches are typically adding large numerical dissipation or dispersion.

Hybrid methods. Many authors proposed to combine the benefits of particle-based and grid-based methods, offering hybrid methods such as MultiFLIP [Boyd and Bridson 2012], Material Point Method (MPM) [Yan et al. 2018; Gao et al. 2018a,b], and the level set method with particles/markers [Losasso et al. 2006; Mihalef et al. 2007; Kim 2010] — but often at higher computational costs. A recent survey [Ren et al. 2018] further details this class of approaches.

2.2 N-phase fluid solvers in CFD

In CFD, the study of incompressible multiphase flows has given rise to several prominent numerical methodologies to deal with fluid interfaces, including the volume of fluid method [Elliott and Luckhaus 1991], the level set method (LSM) [Osher and Sethian 1988; Sussman et al. 1994], the front tracking method [Tryggvason et al. 2001], and the phase field method (PFM) [Anderson et al. 1998; Ding et al. 2007]. Phase-field methods have attracted increasing interest in recent years due to various issues with the other approaches (e.g., loss of volume for LSM, interface reconstruction for VOF, and topology changes for front-tracking, to name a few [Wang et al. 2019]). In particular, researchers have proposed different governing equations, such as the fourth-order Cahn–Hilliard equation (CH) [Cahn and Hilliard 1958], the second-order Allen–Cahn equation (AC) [Allen and Cahn 1976] and the second-order conservative phase field equation (CPF) [Chiu and Lin 2011; Fakhari et al. 2017b]. Both CH and AC rely on a H^{-1}- and L^2-gradient flow of a functional [Bao and Du 2011]. But just like the level set method, CH and AC do not inherently conserve the total mass of the fluid(s) unless significant computational efforts are employed [Rubinstein and Sternberg 1992; Brassel and Bretin 2011]. In contrast, CPF derives its mass-conservative property for each phase through an interface advection equation in divergence form [Chiu and Lin 2011; Sun and Beckermann 2007].

However, phase field methods also face numerical issues in the evolution of the interface to accurately account for the effects of surface tension. Existing approaches include non-monolithic [Lee

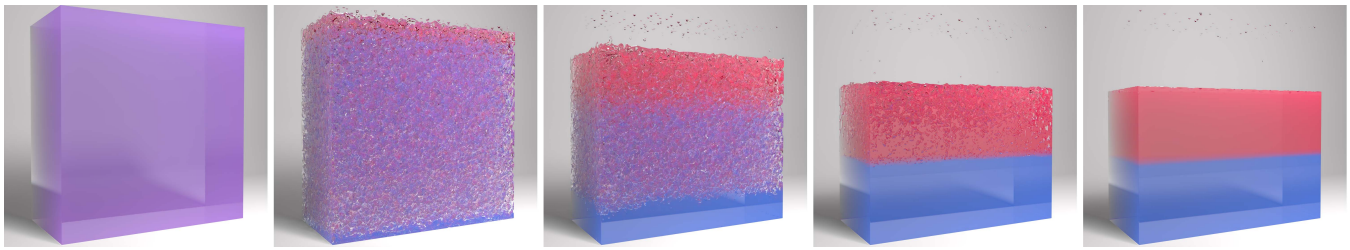


Fig. 3. **Mixture unmixing.** Initializing a three-phase mixture (two liquids, plus air; density ratios: 800 : 80 : 1) with equal volume, the homogeneous mixture gets separated due to gravity. Small droplets are visible on the top wall of the container.

and Kim 2012; Kim and Lee 2017] vs. monolithic models [Wu and Xu 2017; Zheng and Zheng 2019], both built upon the CH equation for solving multiphase flows — the key distinction being whether the phase field equations and surface tension forces are addressed separately or not. Yet, the lack of mass preservation of CH systematically leads to numerical issues [Yue et al. 2007], and incorporating phase field evolution with a LBM fluid solver usually adds further stability issues [Abadi et al. 2018; Fakhari et al. 2017a].

More recently, Hu et al. [2020] proposed a *generalized* conservative phase field model within the LBM framework, which reformulates CPF via the gradient flow of a functional to account for surface tension for n-fluid simulation. Despite all these advancements, state-of-the-art methods still face challenges in handling turbulence and intricate boundary scenarios; in particular, the low accuracy of the Boltzmann collision operator on which they rely does not lead to stable multiphase simulations for moderate to high Reynolds numbers, and their excessive memory usage prevents large 3D simulations.

2.3 Discussion

Based on this overview of previous works, it seems clear that particles, and to a lesser extent, hybrid methods, still have a number of challenges for multiphase fluid simulation: the lack of precise handling of pressure conditions between air and the other phases/fluids seems to prevent a unified treatment of bubble dynamics, splashing, glugging, and wetting. On the other hand, the recent success of Lattice Boltzmann Method (LBM) approaches in graphics, which demonstrated notable gains in both efficiency and accuracy, have managed to capture these types of complex behavior for single- and two-phase flows. Moreover, the phase field method uses a diffuse interface to integrate interfacial forces smoothly across the fluid interfaces, adding stability compared to a sharp interface treatment. These facts motivated our decision to adopt a fully LBM-based approach to offer accurate simulation and guarantee efficiency via a massively parallel implementation. However, the *only method* combining an LBM fluid solver with phase fields to handle multiphase fluid simulation [Hu et al. 2020] has its share of limitations as we reviewed in 2.2 — including only dealing with immiscible fluids and only handling moderate Reynolds numbers.

In the remainder of this paper, we first review the work of Hu et al. [2020] in more detail, before presenting our contributions to extend, generalize, and improve the accuracy, memory usage, and stability of this state-of-the-art method to achieve our goals of efficient and general multiphase fluid simulation.

3 BACKGROUND

We begin our exposition with a brief review of state-of-the-art multiphase methods based on phase fields.

3.1 Phase fields for immiscible flows

When only two immiscible phases are present (typically, air and liquid), a single scalar phase field is needed, for which a value of 0 indicates the presence of one phase while a value of 1 indicates the presence of the other phase. The interface between the two phases is delineated with a hyperbolic tangent steep profile from 0 to 1 over an interfacial thickness ξ, see for instance [Li et al. 2021, 2022; Li and Desbrun 2023]. The case of n immiscible phases or fluids (typically, air and a variety of liquids of different densities and viscosities) is, however, more involved. Each phase must be encoded by its own scalar phase field: the i^{th} phase is a scalar function $\phi_i(\boldsymbol{x}, t)$ where $\boldsymbol{x}$ is a position within the simulation domain Ω and t is time. Similar to the two-phase case, the immiscible phase-field profile at rest is:

$$\phi_i(\boldsymbol{x}, 0) = \frac{1}{2}(1 - \tanh(d_i(\boldsymbol{x})/\xi)), \tag{1}$$

where $d_i(\boldsymbol{x})$ is the distance from $\boldsymbol{x}$ to the interface of the i^{th} phase (defined as $\phi^{-1}(0.5)$), implying that the phase field profile satisfies:

$$|\nabla\phi_i| = \frac{4}{\xi}\phi_i(1 - \phi_i). \tag{2}$$

The set of all order parameters $\boldsymbol{\phi} = (\phi_1, ..., \phi_i, ..., \phi_n)$ then represents n-fluids *iff* they also satisfy:

$$\sum_{i=1}^{n} \phi_i(\boldsymbol{x}, t) = 1 \quad \forall\, \boldsymbol{x} \in \Omega. \tag{3}$$

i.e., the volume fractions of the n phases sum to one everywhere, to ensure they, together, fill up the whole simulation domain. This last constraint allows us to store only $(n-1)$ phase fields for n phases: the remaining phase field (say, ϕ_n) is trivially evaluated on the fly as $\phi_n(\boldsymbol{x}, t) = 1 - \sum_{i=1}^{n-1} \phi_i(\boldsymbol{x}, t)$.

3.2 Generalized phase-field model for immiscible fluids

However, compared to the two-phase case, the sets of constraints described above (Eqs. (2) for $1 \le i \le n$ and Eq. (3)) form an over-constrained system. Recent methods [Dong 2017; Wu and Xu 2017] have thus proposed a least squares method instead. They consider the following functional to minimize, where Eq. (2) is best enforced under the constraint of Eq. (3):

$$W(\boldsymbol{\phi}, \nabla\boldsymbol{\phi}) = \int_\Omega \sum_{i=1}^{n} \left[|\nabla\phi_i| - \frac{4}{\xi}\phi_i\,(1 - \phi_i) \right]^2 + \gamma \int_\Omega \left(\sum_{i=1}^{n} \phi_i \; - \; 1 \right) d\boldsymbol{x},$$

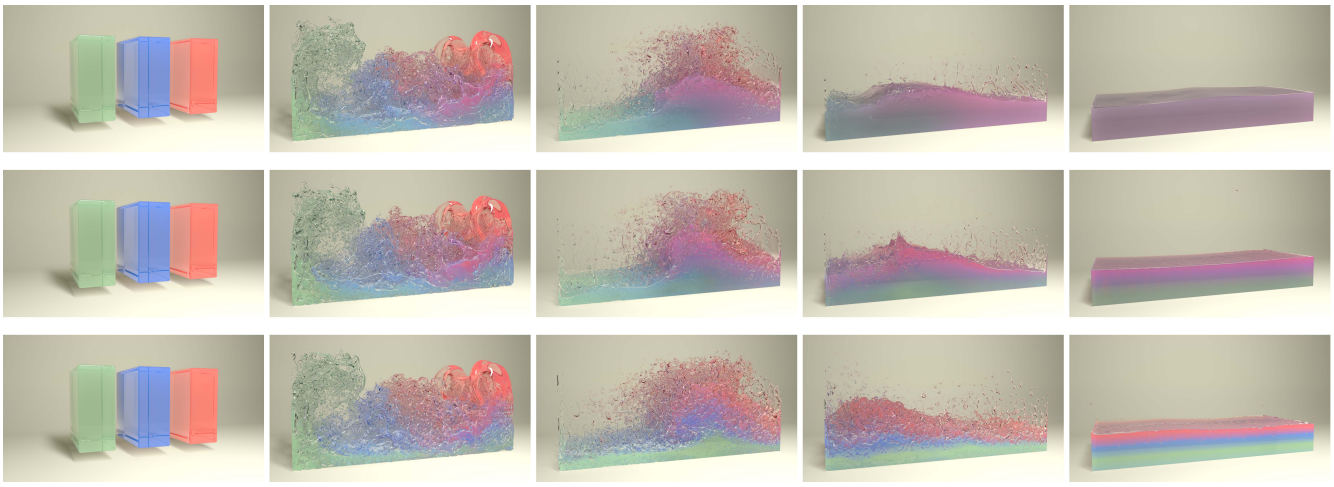


Fig. 4. **Dam break with three miscible/immiscible phases.** Three liquids, immiscible with the air phase, are dropped in a container. When they are all miscible with each other (top), they finally form a grey mixture; For partial miscibility (middle), a resulting half-mixed layer interface appears. If they are all immiscible (bottom), clear interfaces between different liquids remain at the end.

where γ is a Lagrangian multiplier to enforce the constraint. Minimizing W via a L^2-gradient flow leads to the evolution equation:

$$\frac{\partial \phi_i}{\partial t} = -\mu_i \frac{\delta W}{\delta \phi_i}, \tag{4}$$

where $\mu_i > 0$ is the mobility of the i^{th} fluid. If we add an advection of the phase fields in the fluid velocity $\boldsymbol{u}$, we get the *generalized conservative phase field equations for immiscible fluids*:

$$\frac{\partial \phi_i}{\partial t} + \boldsymbol{u} \cdot \nabla \phi_i = \mu_i \left\{ \nabla^2 \phi_i - \frac{1}{\xi} \nabla \cdot \left[4\phi_i (1-\phi_i) \frac{\nabla \phi_i}{|\nabla \phi_i|} \right] \right\} \tag{5}$$
$$- \frac{8\mu_i (2\phi_i - 1)}{\xi} \left[|\nabla \phi_i| - \frac{4}{\xi} \phi_i (1-\phi_i) \right] + \gamma,$$

where ξ is the interfacial thickness. Each phase's evolution equation is therefore similar to the two-phase case from previous LBM papers in graphics, except for the presence of the Lagrangian multiplier γ which we obtain by summing Eqs. (5) for i from 1 to n, yielding:

$$\gamma = \frac{1}{n} \sum_j \mu_i \left(\nabla \cdot \left[\frac{4\phi_j (1-\phi_j)}{\xi} \frac{\nabla \phi_j}{|\nabla \phi_j|} \right] + \frac{8(2\phi_j - 1)}{\xi} \left[|\nabla \phi_j| - \frac{4}{\xi} \phi_j (1-\phi_j) \right] \right).$$

Various authors came to the conclusion that this expression can be simplified without loss of accuracy, and can be made to remain valid in the general case where the number of phases can come and go in the computational domain throughout the simulation. First, the second term can be discarded since this approach is based on making Eq. (2) holds in the least squares sense. Second, to enforce what Dong [2017] and Wu and Xu [2017] call "reduction consistency" (i.e., the evolution equation should be valid for any number of phases being present at a given time), Hu et al. [2020] proposed to replace the single γ by a γ_i per phase, and to replace the $1/n$ term by χ_i/χ where $\chi_i = 1$ (resp., $\chi_i = 0$) indicates that the i^{th} phase is currently present (resp., absent) in the domain of simulation, while $\chi = \sum_i \chi_i$. Indeed, this slightly altered version enforces that χ_i/χ will always be $1/k$ when k fluids are present. Consequently, the following phase field equations were used:

$$\frac{\partial \phi_i}{\partial t} + \boldsymbol{u} \cdot \nabla \phi_i = \mu_i \left\{ \nabla^2 \phi_i - \frac{1}{\xi} \nabla \cdot \left[4\phi_i (1-\phi_i) \frac{\nabla \phi_i}{|\nabla \phi_i|} \right] \right\} + \gamma_i \tag{6}$$

where γ_i is expressed as:

$$\gamma_i = \frac{\chi_i}{\xi \chi} \sum_{j=1}^{n} \mu_i \nabla \cdot \left[4\phi_j (1-\phi_j) \frac{\nabla \phi_j}{|\nabla \phi_j|} \right]; \tag{7}$$

and as mentioned earlier, only $(n-1)$ phases are actually integrated, as the last one is deduced directly using the constraint in Eq. (3).

3.3 Lattice Boltzmann integration of phase field model

To solve Eq. (6) efficiently, Hu et al. [2020] proposed a lattice Boltzmann scheme using a D3Q19 (resp., D2Q9) lattice structure in 3D (resp., 2D). Hence, a distribution function $\boldsymbol{h}^i = \{h_0^i, h_1^i, \dots h_{18}^i\}$ for each phase i is used, from which the phase can be recovered through its zero-th order velocity moment: $\phi_i(\mathbf{x}, t) = \sum_k h_k^i(\mathbf{x}, t)$. Finally, the time evolution of $\boldsymbol{h}^i$ is achieved through a raw-moment multiple-relaxation-times (RM-MRT) model to integrate Eq. (6) in time.

Unfortunately, this LB approach to numerically simulate the generalized conservative phase field model suffers from a number of issues in practice. First and foremost, it requires an unreasonable amount of memory, since the distribution function *for each phase* needs 38 scalar values per node of a fine 3D regular grid. Second, it is only capable of handling immiscible fluid components – a strong limitation for graphics applications. Third, its low-order treatment of the Boltzmann collision terms does not allow for turbulent behaviors of the fluids or large density ratios – two other important limitations for graphics where visual complexity of water-air interaction for instance is often paramount. Finally, Hu et al. [2020] used a (mostly 2D) LBM solver which cannot handle complex boundaries.

3.4 Contributions at a glance

Our work improves on the temporal evolution of phase fields and velocity in multiphase flow simulation by removing the most restrictive limitations of state-of-the-art methods in CFD. We will show that the overly-high memory requirements of [Hu et al. 2020] can be drastically reduced by the use of a D3Q7 lattice structure for phases and a moment-based encoding of the distribution function representing the mesoscopic velocity. We will also derive a new

Fig. 5. **Flows through pumice.** Two liquids with different densities are poured on the top of a porous rock containing a multitude of cavities and tunnels and the heavier liquid (blue) starts flowing slowly through pumice, followed by the lighter one (cyan). The two liquids end up separated at the bottom.

conservative phase field model providing a unified solver for both miscible and immiscible multiphase flows. To improve accuracy (and thus, stability) in the integration of the phase field equations, we will also leverage the two-scale approach proposed in [Li and Desbrun 2023]. Coupled with a fast, accurate, and low-memory-footprint fluid solver, we will demonstrate that our new n-phase solver performs turbulent multiphase simulations efficiently, with improved pressure behavior near solid boundaries.

4 OUR NOVEL LIGHTWEIGHT PHASE-FIELD MODEL

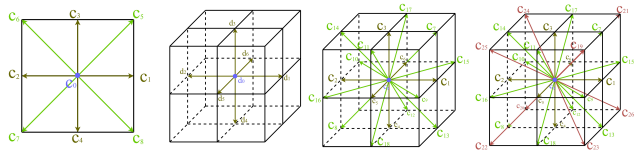
We now introduce our generalized conservative phase field model for (im)miscible flows. Combined with a novel n-phase variant of the recently proposed high-order moment encoded LBM (HOME-LBM) method, we will show that our approach to encode and integrate in time the phase fields needed to simulate multiphase flows removes many of the most stringent shortcomings of [Hu et al. 2020].

Table 1. Summary of the main variables used in our method.

Symbol	Physical meaning
ϕ_i	i^{th} phase indicator
$\boldsymbol{h}^i$	distribution function for i^{th} phase field
μ_i	i^{th} phase mobility
$\boldsymbol{u}$	flow velocity
$\boldsymbol{g}$	distribution function for flow field
ρ	flow density
p	flow pressure
$\mathbf{F}_s$	surface tension force
$\mathbf{F}_v$	viscosity force
$\mathbf{F}_p$	pressure force
$\mathbf{F}_b$	body force

4.1 Unifying immiscible and miscible flows

While Hu et al. [2020] only considered immiscible fluids, we can take a hint from the two-phase case for which the profile is $|\nabla\phi| = \frac{4}{\xi}\phi(1-\phi)$, where the product of one phase (ϕ) and the other phase ($1-\phi$) is directly proportional to the gradient of the phase field. The immiscible case follows the same logic, where now it is the sum of all pairwise phase products, as explicitly stated in Eq. (2) since $1-\phi_i$ is equal to the sum of all other phase fields due to the constraint in Eq. (3). We can then extend these immiscible cases to handle miscible or even partially miscible fluids as we detail next,



(a) (b) (c) (d)

Fig. 6. **Lattice Structures.** In LBM, mesoscopic velocities are discretized using lattice velocities. Our velocity-based distribution $\boldsymbol{g}$ uses a D2Q9 lattice in 2D (a), and a D3Q19 (c) or D3Q27 (d) lattice in 3D. A D3Q7 lattice (b) is used for our phase-field distribution $\boldsymbol{h}$ in 3D.

which matches the recent work of He et al. [2020] in the case of three fluids. First, define pairwise miscibility values $\{\sigma_{ij}\}_{i\neq j}$ where $\sigma_{ij}=1$ means that phases i and j do not mix, while $\sigma_{ij}=0$ means that phases i and j are entirely miscible. (Note that values of σ_{ij} between 0 and 1 can also be used to deal with partial miscibility.) Then, the phase field profile can be re-expressed as:

$$|\nabla\phi_i| = \frac{4}{\xi}\phi_i\Big(\sum_{j=1\&j\neq i}^{n}\sigma_{ij}\phi_j\Big), \tag{8}$$

to unify the case of miscible vs. immiscible fluids. We can now go through the least-squares minimization explained in Sec. 3 for these new profiles to find out that Eq. (6) has now changed into:

$$\frac{\partial\phi_i}{\partial t} + \boldsymbol{u}\cdot\nabla\phi_i = \mu_i\left\{\nabla^2\phi_i - \frac{1}{\xi}\nabla\cdot\left[4\phi_i\Big(\sum_{j=1\&j\neq i}^{n}\sigma_{ij}\phi_j\Big)\frac{\nabla\phi_i}{|\nabla\phi_i|}\right]\right\} + \gamma_i \tag{9}$$

for a Lagrange multiplier γ_i simplified down to:

$$\gamma_i = \frac{\chi_i}{\xi\chi}\sum_{j=1}^{n}\mu_i\nabla\cdot\left[4\phi_j(\sum_{\substack{k=1\\k\neq j}}^{n}\sigma_{jk}\phi_k)\frac{\nabla\phi_j}{|\nabla\phi_j|}\right]. \tag{10}$$

4.2 Integrating phase fields with D3Q7 CM-MRT LBM

At this point, we could use the lattice Boltzmann approach from [Hu et al. 2020] to integrate our unified model from Eq. (8) in order to advance the phase fields in time. However, its use of a D3Q19 lattice structure requires a huge amount of memory, and the raw-moment MRT approximation for the Boltzmann collision terms is known to fail Galilean invariance [d'Humières 2002] and to introduce too much dissipation to properly handle turbulence. Instead, we draw inspiration from Li et al. [2022] and Li and Desbrun [2023] who

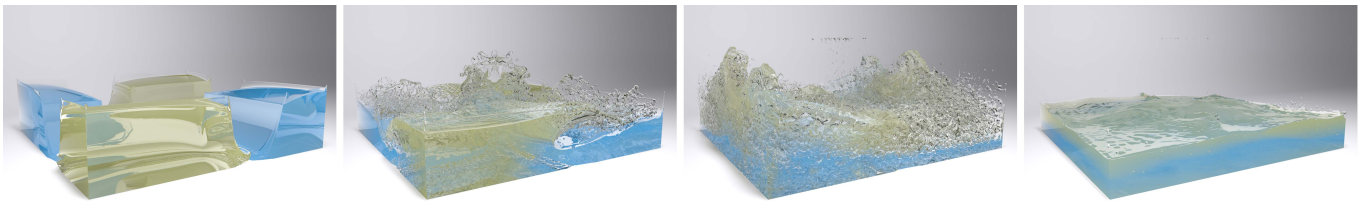


Fig. 7. **Quad dam break with large density ratios.** The density ratios between the three phases (blue phase, cyan phase, and air) are set to 2000:200:1 in this example. Our solver can handle this type of density ratios and Reynolds numbers; we are not aware of other methods able to run this case.

successfully developed a new central-moment multiple-relaxation-times (CM-MRT) model on D3Q7 lattice structure. As their recent approach saves a lot of memory usage and improves the accuracy of collision terms, we extend it to our case of multiple phases.

For each phase field ϕ_i of the $(n-1)$ first phases, we need to integrate in time a seven-velocity distribution $\boldsymbol{h}^i = \{h_0^i, ..., h_6^i\}$ that is stored at each node of the regular grid discretizing the simulation domain. The time evolution of $\boldsymbol{h}^i$ for this LBM setup can then be written as a function of the "equilibrium" distribution function $\bar{\boldsymbol{h}}^i$:

$$\bar{h}_\alpha^i = w_\alpha^d \phi_i \Big(1 + \frac{\boldsymbol{d}_\alpha \cdot \boldsymbol{u}}{d_s^2}\Big), \tag{11}$$

where w_α^d (resp., $\boldsymbol{d}_\alpha$) for $\alpha \in \{0, 1, ..., 5, 6\}$ represents the α^{th} lattice weight (resp., lattice velocity) of our D3Q7 lattice at a given node (see Fig. 6), $\boldsymbol{u}$ is the local macroscopic velocity, and $d_s = \frac{1}{2}$ is the speed of sound for the D3Q7 lattice structure. The update equation in time of each distribution function reads

$$\begin{aligned}\boldsymbol{h}^i(\boldsymbol{x}+\boldsymbol{d}, t+1) = \boldsymbol{h}^i(\boldsymbol{x}, t) - \mathbf{M}_h^{-1}\mathbf{S}_h\mathbf{M}_h\big(\boldsymbol{h}^i(\boldsymbol{x}, t) - \bar{\boldsymbol{h}}^i(\boldsymbol{x}, t)\big) \\ + \mathbf{M}_h^{-1}(1 - \tfrac{1}{2}\mathbf{S}_h^i)\mathbf{M}_h\mathbf{H}^i,\end{aligned} \tag{12}$$

where the matrix $\mathbf{M}_h$ is a matrix to transform the non-equilibrium distribution $\boldsymbol{h}^i - \bar{\boldsymbol{h}}^i$ into central-moment space ($\mathbf{M}_h^{-1}$ performs the inverse transform), while the diagonal matrix $\mathbf{S}_h = \text{diag}(1, 1/(4\mu_i + ½), 1/(4\mu_i + ½), 1/(4\mu_i + ½), 1, 1, 1)$ stores the separate relaxation times of each moment — a key element of MRT models which reduces the instabilities that the single relaxation time of the original BGK model for the LBM collision operator typically generates. Because we use a central-moment space instead of a raw-moment space [Hu et al. 2020] to enforce Galilean invariance and improve accuracy of the collision term, the rows of matrix $\mathbf{M}_h = (\mathrm{M}_h^0, \mathrm{M}_h^1, \mathrm{M}_h^2, \mathrm{M}_h^3, \mathrm{M}_h^4, \mathrm{M}_h^5, \mathrm{M}_h^6)^T$ are defined as:

$$\begin{aligned}\mathrm{M}_h^0 &= (1, 1, 1, 1, 1, 1, 1)^T, \\ \mathrm{M}_h^1 = \overline{d}_{\alpha,x}, \quad \mathrm{M}_h^2 &= \overline{d}_{\alpha,y}, \quad \mathrm{M}_h^3 = \overline{d}_{\alpha,z}, \\ \mathrm{M}_h^4 = \overline{d}_{\alpha,x}^2 - \overline{d}_{\alpha,y}^2, \quad \mathrm{M}_h^5 &= \overline{d}_{\alpha,x}^2 - \overline{d}_{\alpha,z}^2, \\ \mathrm{M}_h^6 &= \overline{d}_{\alpha,x}^2 + \overline{d}_{\alpha,y}^2 + \overline{d}_{\alpha,z}^2.\end{aligned} \tag{13}$$

where $\overline{d}_{\alpha,\delta} = \big(\boldsymbol{d}_\alpha - \boldsymbol{u}\big)_\delta$ for lattice index $\alpha \in \{0, ..., 6\}$ and for $\delta \in \{x, y, z\}$ representing a vector coordinate.

Finally the source term $\mathbf{H}^i$ (last term in Eq. (12)), once adapted from [Chai et al. 2018], is expressed as:

$$\begin{aligned}\mathbf{H}_\alpha^i = &\frac{4\phi_i\Big(\sum_{j=1\&j\neq i}^n \sigma_{ij}\phi_j\Big)}{\xi} w_\alpha^d \ \boldsymbol{d}_\alpha \cdot \frac{\nabla\phi_i}{|\nabla\phi_i|} \\ &- \frac{\chi_i}{\xi\chi}\sum_{j=1}^n 4\phi_j\Big(\sum_{k=1\&k\neq j}^n \sigma_{jk}\phi_k\Big) w_\alpha^d \ \boldsymbol{d}_\alpha \cdot \frac{\nabla\phi_j}{|\nabla\phi_j|} \cdot\end{aligned} \tag{14}$$

Each ϕ_i is then updated through the zeroth order moment of its associated distribution function $\boldsymbol{h}^i$, except for the last one which is directly deduced to enforce the constraint from Eq. (3), i.e.,

$$\phi_i = \begin{cases} \sum_\alpha h_\alpha^i & \text{if } 1 \le i \le n-1 \\ 1 - \sum_{j=1}^{n-1}\phi_j & \text{if } i = n \end{cases} \tag{15}$$

Note that we adopt the isotropic centered difference schemes [Li et al. 2021, 2022] for the evaluation of $\nabla\phi_i$ and $\nabla^2\phi_i$: they are computed respectively as:

$$\nabla\phi_i = 3\sum_\alpha \boldsymbol{d}_\alpha w_\alpha^d \phi_i(\boldsymbol{x} + \boldsymbol{d}_\alpha, t); \tag{16}$$

$$\nabla^2\phi_i = 6\sum_\alpha w_\alpha^d \left[\phi_i(\boldsymbol{x} + \boldsymbol{d}_\alpha, t) - \phi_i(\boldsymbol{x}, t)\right], \tag{17}$$

except for the n^{th} phase where Eq. (15) implies:

$$\nabla\phi_n = -\sum_{i=1}^{n-1}\nabla\phi_i\,, \qquad \nabla^2\phi_n = -\sum_{i=1}^{n-1}\nabla^2\phi_i. \tag{18}$$

5 ADAPTING HOME-LBM FOR MULTIPHASE FLOWS

Now that we have detailed how phases are handled, we discuss how to simulate the flow part, i.e., the evolving velocity field (which we will encode via a distribution $\mathbf{g}$) in which the phases are being advected. We will proceed in three steps: first, we will review the macroscopic model typically used for multiphase flow; then we will explain how the HOME-LBM approach of [Li et al. 2023] can be modified not only to save memory, but also to accelerate computations and bring stability by avoiding spurious oscillations near obstacles; finally, we will provide the modified equations for distribution reconstruction and collision terms which are at the core of our contributions.

5.1 Macroscopic model and its mesoscopic discretization

The momentum and continuity equations for incompressible multiphase flows are often described macroscopically as:

$$\begin{cases} \partial\rho/\partial t + \nabla\cdot(\rho\boldsymbol{u}) = 0, & \text{(19a)} \\ \partial(\rho\boldsymbol{u})/\partial t + \nabla\cdot(\rho\,\boldsymbol{u}\otimes\boldsymbol{u}) = -\nabla p + \nabla\cdot\Pi + \mathbf{F}_s + \mathbf{F}_b, & \text{(19b)} \\ \nabla\cdot\boldsymbol{u} = 0, & \text{(19c)} \end{cases}$$

where ρ and $\boldsymbol{u}$ are respectively the spatially-varying fluid density and velocity of the fluids, p is the hydrodynamic pressure enforcing the incompressibility condition in Eq. (19c), and Π is the viscous stress tensor, while $\mathbf{F}_b$ and $\mathbf{F}_s$ are body and surface tension forces. Note that in our multiphase case, the density ρ is written in terms

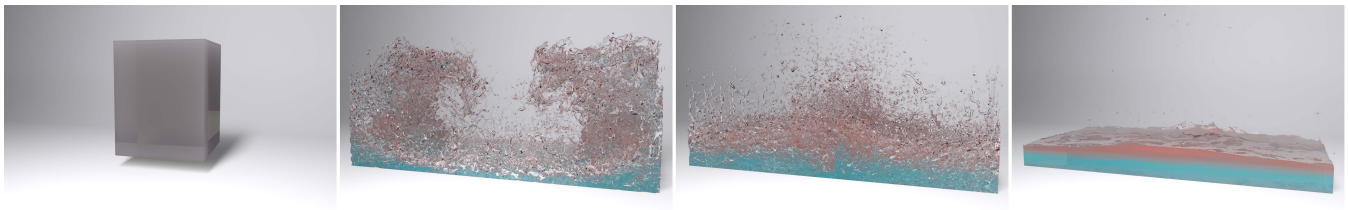


Fig. 8. **Mixture splashing.** A mixture is dropped in a container, forming splashes, turbulence, and bubbles, before becoming separated in layers.

of the densities ρ_i and phase fields ϕ_i of all fluids as:

$$\rho(\boldsymbol{x}) = \sum_{i=1}^{n-1} \rho_i \phi_i(\boldsymbol{x}) + \rho_n \big(1 - \sum_{i=1}^{n-1} \phi_i(\boldsymbol{x})\big), \tag{20}$$

where the last term uses the phase field ϕ_n as defined in Eq. (15); consequently, its gradient is: $\nabla\rho(\boldsymbol{x}) = \sum_{i=1}^{n-1}(\rho_i - \rho_n)\nabla\phi_i(\boldsymbol{x})$.

Alternative formulation. Another formulation of the momentum equation (19b) is worth noting: we can turn all the terms containing the density ρ explicitly as either new variables or external forces through the chain rule [Fakhari et al. 2017b], leading to

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla\cdot\left(\boldsymbol{u}\otimes\boldsymbol{u} + p^* c_s^2 \boldsymbol{I} - [\nu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T)]\right) = \frac{1}{\rho}(\mathbf{F}_p + \mathbf{F}_\nu + \mathbf{F}_s + \mathbf{F}_b) \tag{21}$$

where three new terms involve ρ implicitly: $p^* = p/(\rho c_s^2)$ is the "normalized" pressure, $\mathbf{F}_p$ is a pressure force, and $\mathbf{F}_\nu$ is a viscosity force, with

$$\mathbf{F}_p = -p^*\,\nabla\rho, \tag{22}$$

$$\mathbf{F}_\nu = [\nu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T)]\nabla\rho, \tag{23}$$

the multi-fluid kinematic viscosity ν being written as a function of each phase viscosity ν_i as: $\nu = \big(\frac{1}{\nu_n} + \sum_i^{n-1}(\frac{1}{\nu_i} - \frac{1}{\nu_n})\big)^{-1}$. The value of this "velocity-based" formulation in Eq. (21) is explained next.

Velocity-based distribution function. Most early LBM works in graphics and CFD were based on a lattice-Boltzmann discretization of Eq. (19) using a distribution function $\boldsymbol{f}$, whose zeroth-order velocity moment being the mass density ρ — considered constant since they were targeting single-phase fluid simulation. For the case of spatially-varying ρ with high-density ratios between phases, it was noticed that mixing $\boldsymbol{u}$ with ρ in the momentum equation was bringing numerical instability due to the rapid changes in density and thus in momentum, so discretizing Eq. (21) instead was proposed [Fakhari et al. 2017b]: this allows for the velocity update and phase-field update to be mostly decoupled, except for the force terms based on the phase fields which influence the motion of the fluids and for the motion of the flow that carries the phase fields along. This is precisely what was proposed recently in graphics [Li et al. 2022; Li and Desbrun 2023], using a different distribution function labeled $\boldsymbol{g}$ here to differentiate from the distribution $\boldsymbol{f}$ of traditional LBM methods: their "velocity-based" LBM discretization for incompressible two-phase flows ended up being formulated as:

$$g_i(\boldsymbol{x} + \mathbf{c}_i, t+1) = g_i(\boldsymbol{x}, t) + \Omega_i^g(\boldsymbol{x}, t) + G_i(\boldsymbol{x}, t), \tag{24}$$

where c_i is a lattice velocity (see Fig. 6) and Ω_i^g and G_i are the influences on the distribution $\boldsymbol{g}$ from the collision operator and the external forces respectively. Macroscopic variables used in Eq. (21) can be recovered from the first three moments of distribution $\boldsymbol{g}$:

$$p^* = \sum_{i=0}^{q-1} g_i, \quad \boldsymbol{u} = \sum_{i=0}^{q-1} \boldsymbol{c}_i g_i + \frac{1}{2}\mathbf{F}, \quad S_{\alpha\beta} = \sum_{i=0}^{q-1} (\boldsymbol{c}_i^2 - \frac{1}{3}\delta_{\alpha\beta})\, g_i\,, \tag{25}$$

where δ is the Kronecker delta function and $\mathbf{F}$ is the sum of external forces, while $S_{\alpha\beta}$ denotes the component of tensor $\boldsymbol{S}$ defined as

$$\boldsymbol{S} = \boldsymbol{\Pi} - \frac{1}{3}p^* = \boldsymbol{u}\otimes\boldsymbol{u} - \nu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T). \tag{26}$$

5.2 Adapting HOME-LBM for multiphase simulation

In the context of monophase fluid simulation, Li et al. [2023] recently proposed a very compact storage of distribution functions $\boldsymbol{f}$ through only their first three velocity moments ρ, $\rho\boldsymbol{u}$, and $\rho\boldsymbol{S}$ at each grid node instead of the discretization of $\boldsymbol{f}$ in 27 directions $\{f_i\}_{i=0..26}$ — saving nearly two third of the memory footprint. This *HOME-LBM* approach offers a high-order reconstruction of the distribution from its three moments to filter out non-physical oscillations in turbulent flow and near complex boundaries, as well as a moment-based collision model. Our use of a different type of "velocity-based" LBM discretization (based on Eq. (21) instead of Eq. (19b)) requires important alterations to their approach to adapt it to our multiphase context, but it will improve boundary pressure accuracy, accelerate performance, and reduce memory use for our solver. In the remainder of this section, we will assume a D3Q27 lattice discretization in 3D as typically recommended for turbulent flows to ensure numerical accuracy; we will also provide all necessary expressions for D3Q19-based distribution functions to further reduce computations with only minor accuracy loss for n-phase flow simulation.

5.2.1 Modifying distribution reconstruction from moments. Distribution functions $\boldsymbol{f}(\boldsymbol{v}, \boldsymbol{x}, t)$ in LBM are typically encoded (before discretization) through truncated Hermite series expansions of the mesoscopic velocity v [Shan et al. 2006]. In 3D, the mesoscopic velocity is discretized through D3Q27 lattice velocities $\{\boldsymbol{c}_i\}_{i=1..27}$ (see Fig. 6), the continuous function $\boldsymbol{f}(\boldsymbol{v}, \boldsymbol{x}, t)$ is thus turned into 27-value vectors $\boldsymbol{f}(\boldsymbol{x}, t) = \{f_i\}_{i=0..26}$ per node $\boldsymbol{x}$ of a regular grid and per discrete time t, through $f_i(\boldsymbol{x}, t) = w_i f(\boldsymbol{c}_i, \boldsymbol{x}, t)/\omega(\boldsymbol{c}_i)$, where w_i are Gauss–Hermite quadrature weights for the D3Q27 lattice while $\omega(\boldsymbol{v}) \propto \exp(-\|\boldsymbol{v}\|^2/2)$ is a normalized weighting function (see, for instance, [Li et al. 2020] for detailed derivations). For our n-phase case, the change of main variables proposed in [Fakhari et al. 2017b] leads to a new distribution function g discretized as $\boldsymbol{g} = \{g_i\}_{i=0..26}$ on a regular grid and at discrete times, now defined as

$$g_i(\boldsymbol{x}, t) = w_i\, g(\boldsymbol{c}_i, \boldsymbol{x}, t)/\omega(\boldsymbol{c}_i) = \frac{f_i(\boldsymbol{x}, t)}{\rho(\boldsymbol{x}, t)} + w_i\,(p^*(\boldsymbol{x}, t) - 1). \tag{27}$$

Recall that in HOME-LBM, the encoding of the discrete distribution function $\boldsymbol{f}$ was done through only its first three velocity moments,

i.e., ρ, $\rho\boldsymbol{u}$, and $\rho\boldsymbol{S}$ [Li et al. 2023]. From Eq. (27), we can instead encode the distribution $\boldsymbol{g}$ by storing its first three moments p^* (scalar), $\boldsymbol{u}$ (vector), and $\boldsymbol{S}$ (symmetric tensor) defined in Eq. (25) — thus reducing the 27 components g_i into 10 values.

Given the three moments stored in our HOME-LBM for multiphase flows, we follow [Li et al. 2023] and derive through Mathematica a new third-order Hermite-based "filtered" reconstruction of the distribution components g_i from the velocity moments p^*, $\boldsymbol{u}$, and $\boldsymbol{S}$. The resulting expression is:

$$\begin{aligned} g_i =& w_i\Big[p^* + \frac{\boldsymbol{c}_i\cdot\boldsymbol{u}}{c_s^2} + \frac{\boldsymbol{H}^{[2]}(\boldsymbol{c}_i):\boldsymbol{S}}{2c_s^4} \\ &+ \tfrac{1}{2c_s^6}\big(\quad H^{[3]}_{xxy}(\boldsymbol{c}_i)(S_{xx}u_y + 2S_{xy}u_x - 2u_xu_xu_y) \\ &\quad + H^{[3]}_{xyy}(\boldsymbol{c}_i)(S_{yy}u_x + 2S_{xy}u_y - 2u_xu_yu_y) \\ &\quad + H^{[3]}_{xxz}(\boldsymbol{c}_i)(S_{xx}u_z + 2S_{xz}u_x - 2u_xu_xu_z) \\ &\quad + H^{[3]}_{xzz}(\boldsymbol{c}_i)(S_{zz}u_x + 2S_{xz}u_z - 2u_xu_zu_z) \\ &\quad + H^{[3]}_{yzz}(\boldsymbol{c}_i)(S_{zz}u_y + 2S_{yz}u_z - 2u_yu_zu_z) \\ &\quad + H^{[3]}_{yyz}(\boldsymbol{c}_i)(S_{yy}u_z + 2S_{yz}u_z - 2u_yu_yu_z) \\ &\quad + H^{[3]}_{xyz}(\boldsymbol{c}_i)(S_{xz}u_y + S_{yz}u_x + S_{xy}u_z - 2u_xu_yu_z)\big)\Big]. \end{aligned} \tag{28}$$

where $\boldsymbol{H}$ are the Hermite tensors given in App. A. We also provide the reconstruction expression for D3Q19 in App. D, as it reduces both memory requirements and computational cost for the reconstruction, without a noticeable loss in accuracy or stability.

5.2.2 HOME collision model. Once the three moments have been converted back into a full set of distribution functions as described above, we could now proceed as typically done in a LBM method: integrating Eq. (24), the evolution equation of $\boldsymbol{g}$, through Strang splitting via streaming, followed by collision evaluation, and finally external force injection, before saving the new three moments for all grid nodes after integration. However, Li et al. [2023] proposed instead to convert the resulting streamed distribution back to its three first moments *right after streaming* using Eqs. (25); they then showed how to explicitly add high-order collision terms and external forces *directly* to these three moments to complete one integration timestep. We follow exactly their HOME-based approach, adapted to our new distribution $\boldsymbol{g}$. That is, we first perform the regular streaming process; the new three moments are then evaluated through Eqs. (25) and we denote them temporarily $p^{*'}$, $\boldsymbol{u}'$, and $\boldsymbol{S}'$ as they represent the moments *before* their final alteration by collision and external forces. Then, we perform the sixth-order Hermite expansion of the equilibrium central-moments and external forces recommended in [Li et al. 2023] and compute the update of the three first moments due to the effect of the (multiple-relaxation rate) collision and forces in closed form. Using Mathematica [Wolfram Research Inc. 2023], and for $\tau = 3\nu + 0.5$, we found that the closed-form updates read:

$$p^*(\boldsymbol{x}, t+1) = p^{*'}; \tag{29}$$

$$u_\alpha(\boldsymbol{x}, t+1) = u'_\alpha + \frac{1}{2\rho'}F_\alpha; \tag{30}$$

$$S_{xy}(\boldsymbol{x}, t+1) = (1-\tfrac{1}{\tau})S'_{xy} + \tfrac{1}{\tau}u'_xu'_y + \tfrac{2\tau-1}{2\tau\rho'}(F_xu'_y + F_yu'_x); \tag{31}$$

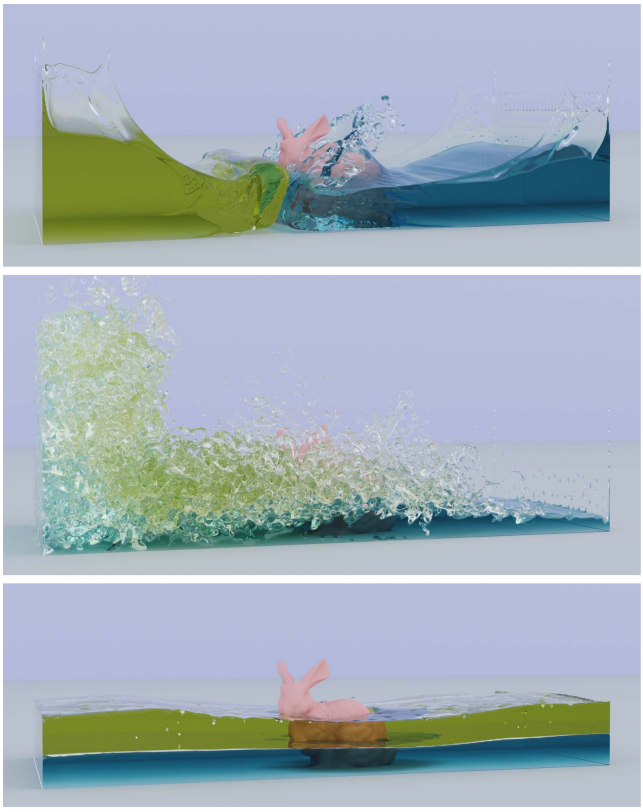


Fig. 9. **Dam breaking over bunny.** Two different liquids in a dam break come splashing onto a bunny-shape obstacle. Small droplets are seen on the bunny due to a hydrophobic wetting boundary condition.

$$\begin{aligned} S_{xx}(\boldsymbol{x}, t+1) =& \tfrac{\tau-1}{3\tau}(2S'_{xx} - S'_{yy} - S'_{zz}) \\ &+ \tfrac{1}{3}\left(u_x'^2 + u_y'^2 + u_z'^2\right) + \tfrac{1}{3\tau}\left(2u_x'^2 - u_y'^2 - u_z'^2\right) \\ &+ \tfrac{1}{\rho'}F_xu'_x + \tfrac{\tau-1}{3\tau\rho'}(2F_xu'_x - F_yu'_y - F_zu'_z); \\ S_{yy}(\boldsymbol{x}, t+1) =& \tfrac{\tau-1}{3\tau}(2S'_{yy} - S'_{xx} - S'_{zz}) \\ &+ \tfrac{1}{3}\left(u_x'^2 + u_y'^2 + u_z'^2\right) + \tfrac{1}{3\tau}\left(2u_y'^2 - u_x'^2 - u_z'^2\right) \\ &+ \tfrac{1}{\rho'}F_yu'_y + \tfrac{\tau-1}{3\tau\rho'}(2F_yu'_y - F_xu'_x - F_zu'_z); \\ S_{zz}(\boldsymbol{x}, t+1) =& \tfrac{\tau-1}{3\tau}(2S'_{zz} - S'_{xx} - S'_{yy}) \\ &+ \tfrac{1}{3}\left(u_x'^2 + u_y'^2 + u_z'^2\right) + \tfrac{1}{3\tau}\left(2u_z'^2 - u_x'^2 - u_y'^2\right) \\ &+ \tfrac{1}{\rho'}F_zu'_z + \tfrac{\tau-1}{3\tau\rho'}(2F_zu'_z - F_xu'_x - F_yu'_y). \end{aligned}$$

Note that these expressions do not change if a D3Q19 discretization is used instead. With these new update equations given above, we avoid a large amount of computations since we do not need to perform the expensive conversion from distribution space to moment space needed in current state-of-the-art collision models (i.e., the equivalent of Eq. (12) but now for a much larger D3Q19 or D3Q27 discretization), without losing accuracy.

5.3 Surface tension

While three of the four forces present in Eq. (21) are simple to evaluate (concretely, $\mathbf{F}_p$ is computed via Eq. (22), $\mathbf{F}_\nu$ via Eq. (23),

and $\mathbf{F}_b$ depends on forces imposed on the fluid), the surface tension force $\mathbf{F}_s$ needs care in our multiphase flow context. We base our evaluation on the work of He et al. [2020] who proposed a unified total free energy $\mathcal{F}$ for the case of ternary fluids. We simply extend their formulation for the n-phase case through:

$$\mathcal{F}(\boldsymbol{\phi}, \nabla\boldsymbol{\phi}) = \int_\Omega \frac{3\xi}{4}\sum_{i=1}^{n}\beta_i|\nabla\phi_i|^2 + \frac{12}{\xi}\sum_{i=1}^{n}\beta_i\phi_i^2\Big(\sum_{j\neq i}\sigma_{ij}\phi_j\Big)^2 d\boldsymbol{x}, \quad (32)$$

where β_i is the capillary coefficient for phase i. Using the virtual work method [Hu et al. 2020], we have

$$\delta\mathcal{F} = F(\boldsymbol{\phi}+\delta\boldsymbol{\phi}, \nabla\boldsymbol{\phi}+\nabla\delta\boldsymbol{\phi}) - F(\boldsymbol{\phi}, \nabla\boldsymbol{\phi}) = -\int_\Omega \mathbf{F}_s\delta\boldsymbol{x}\, d\boldsymbol{x}, \quad (33)$$

where $\delta\boldsymbol{x}$ is a virtual displacement. As a result, the surface tension force can be written as:

$$\begin{aligned}\mathbf{F}_s = \sum_{i=1}^{n}\Big\{&\beta_i\nabla\phi_i\Big[\frac{24}{\xi}\phi_i\Big(\sum_{j\neq i}\sigma_{ij}\phi_j\Big)^2 - \frac{3\xi}{2}\nabla^2\phi_i\Big] \\ &+ \frac{24}{\xi}\beta_i\phi_i^2\Big(\sum_{j\neq i}\sigma_{ij}\phi_j\Big)\Big(\sum_{j\neq i}\sigma_{ij}\nabla\phi_j\Big)\Big\}.\end{aligned} \quad (34)$$

Note that for immiscible fluids, surface tension simplifies down to:

$$\mathbf{F}_s = \sum_{i=1}^{n}\beta_i\nabla\phi_i\Big\{\frac{24}{\xi}\phi_i(1-\phi_i)(1-2\phi_i) - \frac{3\xi}{2}\nabla^2\phi_i\Big\}. \quad (35)$$

5.4 In/outlet and one-way coupling boundary conditions

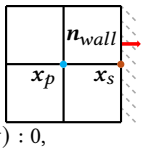
To simulate inlet boundary conditions, we simply set the initial velocity moments of the flow based on the desired pressure p^0 and velocity $\boldsymbol{u}^0$ of the fluid, and set the second-order moment as

$$\boldsymbol{S}^0_{\alpha\beta} = \boldsymbol{u}^0_\alpha\boldsymbol{u}^0_\beta - \frac{1}{3}\delta_{\alpha\beta}, \quad (36)$$

where δ is the Kronecker delta function. To set the phase values ϕ_i^0 for the i^{th} phase, we initialize the distribution function $\mathbf{h}$ as

$$\bar{h}^{i,0}_\alpha = w^d_\alpha\phi^0_i\Big(1 + \frac{\mathbf{d}_\alpha\cdot\boldsymbol{u}^0}{d_s^2}\Big) \quad (37)$$

based on Eq. (11). For outlet boundary conditions, we apply Neumann boundary conditions if the flux through an outlet wall (see inset) is going outwards, or simply set the velocity to zero, i.e.,



$$\begin{aligned}&\boldsymbol{u}_x(\boldsymbol{x}_s, t) = \nabla\boldsymbol{u}(\boldsymbol{x}_p, t)\cdot\boldsymbol{n}_{wall} > 0\ ?\ \boldsymbol{u}_x(\boldsymbol{x}_p, t) : 0,\\ &\boldsymbol{u}_y(\boldsymbol{x}_s, t) = 0,\end{aligned} \quad (38)$$

from which the second-order moment is again approximated as in Eq. (36). For the phase field equation, we implement an outlet boundary condition for the distribution function $\mathbf{h}$ on the wall based on [Lou et al. 2013], written as:

$$\boldsymbol{h}(\boldsymbol{x}_s, t) = \boldsymbol{h}(\boldsymbol{x}_p, t). \quad (39)$$

6 PUTTING IT ALL TOGETHER

We can now complete the description of our HOME-LBM based simulation of multiphase flows by discussing how the solvers for respectively the phases and the flow are coupled.

We have reviewed in Sec. 4 how to deal with the LBM-based n phase fields, indicating where the n fluids are located in the domain. Note that the only other field they require to be integrated in time is the macroscopic velocity field in which they are advected. Conversely, we have discussed in Sec. 5 how to use a lightweight and efficient flow integration to compute a multiphase version of HOME-LBM using a velocity-based distribution to handle the velocity field through LBM. This time, forces based on the phase densities (hence, on all the phases) are needed to properly integrate the velocity field. We thus have all we need to couple the simulation of the n phases $\{\phi_i\}_n$ and the flow simulator to integrate multifluid flows in time.

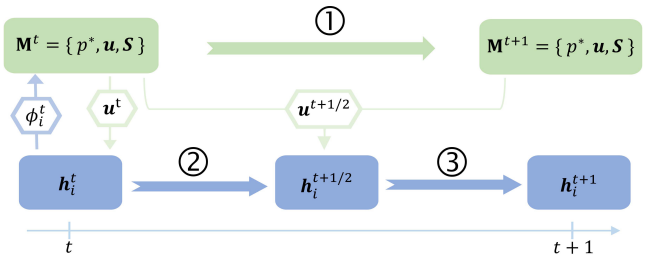


Fig. 10. **Time integration.** A timestep from t to t+1 of our solver is achieved through (1) first, advancing the moments $\mathbf{M}^t$ in time to $t+1$ accounting for the forces induced by the n phase fields ϕ_i^t known at time t (top); then (2) the phase fields are updated in two substeps (bottom), first advected by the macroscopic velocity $\boldsymbol{u}^t$, then by the average velocity $\boldsymbol{u}^{t+\frac{1}{2}}$. Hexagonal labels indicate dependencies between flow and phase-field equations.

However, we added a few other components to this basic simulation framework. First, to improve the accuracy and stability of the phase field equations, we follow the recommendation in [Li and Desbrun 2023] by employing a *eight-times finer resolution for the phase fields* compared to the grid for distribution $\boldsymbol{g}$ (i.e., each spatial dimension uses a twice finer discretization). This changes slightly the time integration scheme since the phase fields now need two substeps, see Fig. 10. Moreover, while Li and Desbrun [2023] had the two grids staggered with respect to each other to filter spurious oscillations of the velocity field, we found that the filtering brought by the use of a HOME-LBM solver (based only on the first three moments) is enough, and no staggering is required. So we kept the two grids aligned as this reduces the need for grid interpolation, hence accelerating the final solver.

Second, we also incorporate wetting (i.e, the way fluids cling to or separate from obstacles) using the same treatment as in two-phase wetting [Li et al. 2022], for all the phases individually — i.e., the boundary conditions to induce hydrophobic or hydrophilic wetting near solid boundaries are set phase by phase, ignoring the fact that phases are globally bound by Eq. (3). While this may be inaccurate, we could not find modeling papers discussing how to properly deal with wetting for multiple phases/fluids. As we will show in Sec. 7, this does not appear to affect the visual validity of our results.

Finally, coupling with complex solids is handled through cut cells, exactly as proposed in [Li and Desbrun 2023], and so do wall boundary conditions. However, we no longer need their "hybrid" bounce-back approach, which was designed to filter the pressure near solids: our use of HOME-LBM removes this issue, and we use the original (simpler) bounce-back approach which results in non-oscillatory pressure behavior near boundaries in our tests. To demonstrate our solver's stability in the presence of large density ratios, we ran a two-phase simulation (a dam break with a thin plate obstacle) for a density ratio of 2000 using different flow models. As Fig. 18 shows, when a solid boundary bounce-back treatment and phase solver are used, the momentum-based LBM model [Fakhari

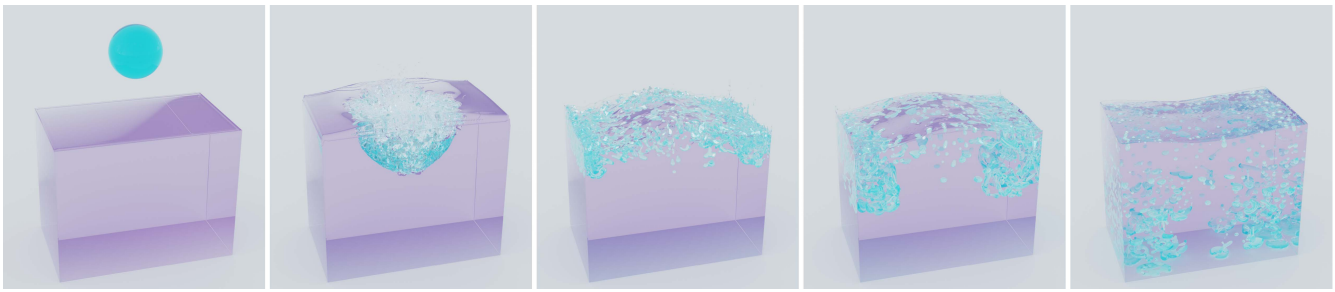


Fig. 11. **Water poured into oil.** With density ratios of water:oil:air set to 800:720:1, a ball of water drops into an oil tank, creating an emulsion.

et al. 2017a] will crash quickly before even hitting the obstacle, while a velocity-based LBM model [Fakhari et al. 2017b] fails when the heavy fluid encounters the obstacle. Only our solver can simulate this case in a stable manner.

7 RESULTS

We now provide ample evidence of the advantages of the resulting multiphase fluid integrator, for which we give pseudocode in Alg. 1. We cover a number of simulation experiments with our fluid solver, including benchmark evaluations, comparisons with existing methods, and several application scenarios. All results were run on a workstation equipped with an AMD Ryzen 9 7900X3D 12-core processor and an Nvidia GeForce RTX 4090 card with 24GB of GPU memory. Our framework was implemented in C++ and CUDA. Detailed statistics, including timings profiled with Nvidia Nsight, are presented in Tab. 3. Note that our handling of one-way coupling relies entirely on [Li and Desbrun 2023], which was developed for two-phase simulations.

GPU-based implementation. We implemented our approach on GPU using a structure-of-arrays (SOA) data structure [Chen et al. 2022]. To accelerate time integration, we maintain two buffers with 10 variables per grid node (representing the first velocity moments of distribution $\boldsymbol{g}$), totaling 20 variables for velocity updates. We also use SOA for each of the $(n-1)$ distributions $\boldsymbol{h}^i$, and we prefetch phase values (from global to GPU shared memory) to further speed up performance for, e.g., the evaluation of phase gradients.

Lightweight distribution reconstruction. For the 3D distribution reconstruction of g_i in Eq. (28), we can also use a lightweight D3Q19 lattice structure that ignores the cube diagonals (see Fig. 6), and for which the distribution reconstruction is given in App. D: by removing the diagonal lattice components, our solver does not lose obvious accuracy because the effect of the collision on the moments stays the same, but the reduction in GPU memory access and in the size of the closed-form evaluation results in higher performance. We thus use this simpler g_i reconstruction for all experiments in this paper.

Memory reduction. Compared to recent work in n-phase simulation [Hu et al. 2020], our framework significantly reduces memory consumption, especially when the phase count n is large. This improvement can be attributed to several factors. First, our framework employs (two times) 10 moment values for velocity discretization, instead of the (two times) 27 distribution function values used in the conventional LBM framework. Secondly, our high-order collision model for phases enables the use of a D3Q7 lattice structure for $(n-1)$ phase equations without compromising accuracy and robustness, while Hu et al. [2020] relied on a D3Q19 lattice, incurring an almost three-fold higher memory cost. Additionally, the decoupling of resolution between flow and phase fields allows us to use a lower resolution for the velocity integration, resulting in substantial memory savings. As shown in Tab. 2, our solver only uses $(12n - 8.5)$ values per grid node to simulate a 3D n-phase flow, while their method requires $(43n + 20)$ variables. For example, in Fig. 4 with $n=4$, our solver offers an approximately 80% reduction in memory size compared to the earlier work of Hu et al. [2020].

ALGORITHM 1: Pseudocode of our kinetic multifluid solver.

```
t ← 0;
Initialize φ, as well as the velocity moments p, u,and S;
Init∇φ()                                          ▷ Eq. (16) and (18)
InitForces()                                      ▷ Eq. (22),(23) and (34)
Initialize distribution h                         ▷ Eq. (11)
while t < T do
    In/OutletBoundaryTreatmentForFlow()           ▷ Sec. 5.4
    HOME-LBM-FlowSolver()                         ▷ Sec. 5.2
    k ← 0;
    while k < 2 do
        PerformPhaseCollision()                   ▷ Eq. (12)
        In/OutletBoundaryTreatmentForPhase()      ▷ Sec. 5.4
        PerformPhaseStream()                      ▷ Eq. (12)
        UpdatePhaseField()                        ▷ Eq. (15)
        WettingBoundaryConditionForPhase()        ▷ Sec. 6
        Compute∇φ()                               ▷ Eq. (16) and (18)
        k ← k + 1;
    end
    CalculateForces()                             ▷ Eq. (22),(23) and (34)
    t ← t + 1;
end
```

Table 2. **Memory reduction:** Number of floats per grid node required in 3D for a n-phase simulation, compared to [Hu et al. 2020]. Uneven numbers account for flow grid nodes being 8 times fewer than phase grid nodes; here $\mathbf{M} = \{p^*, \boldsymbol{u}, \boldsymbol{S}\}$, and $\boldsymbol{n}$ is the phase normal stored for efficiency.

Method	$\mathbf{M}$	$(\rho, \boldsymbol{u}, p^*)$	cut-cell	flag	$\boldsymbol{g}$	F	$\boldsymbol{h}$	ϕ & $\Delta\phi$	$\boldsymbol{n}$	total
Our method	2.5	–	0.5	0.5	–	–	7(n-1)	2(n-1)	3(n-1)	12n-8.5
[Hu et al. 2020]	0	5	0.5	0.5	54	3	38(n-1)	2(n-1)	3(n-1)	43n+20

7.1 Benchmark evaluations

We first perform two classical 2D benchmark tests to verify the correctness of our method.

Spinodal decomposition. Spinodal decomposition occurs when a homogeneous multifluid system undergoes a sudden cooling below its critical temperature, leading to concentration fluctuations. This process, extensively explored in multiphase simulations, separates the system into distinct spatial regions, each rich in one fluid and poor in others. Our test follows the setup of [Abadi et al. 2018], using three fluids with density ratios $\rho_3 : \rho_2 : \rho_1 = 1000 : 100 : 1$ in a $L_0 \times L_0$ simulation domain (for $L_0 = 200$) with periodic boundary conditions. Viscosity and mobility for all fluids are $\nu = 0.5$ and $\mu = 0.8$, and the interface width ξ is set to 4. The initial conditions are:

$$\begin{aligned}
\phi_1(\boldsymbol{x}, 0) &= \bar{\phi}_1 + 0.01 * \text{rand}(), \\
\phi_2(\boldsymbol{x}, 0) &= \bar{\phi}_2 + 0.01 * \text{rand}(), \\
\phi_3(\boldsymbol{x}, 0) &= 1 - \phi_1(\boldsymbol{x}, 0) - \phi_2(\boldsymbol{x}, 0), \\
\boldsymbol{u}(\boldsymbol{x}, 0) &= 0,
\end{aligned}$$

where $\bar{\phi}_1$ and $\bar{\phi}_2$ represent the initial mean concentrations of ϕ_1 and ϕ_2, respectively, while calls to the function $rand()$ generate uniform random numbers within the range $[-1, 1]$. We test the temporal evolution of the three-fluid system with three different sets of mean concentrations: $(\bar{\phi}_1, \bar{\phi}_2) = (\frac{2}{5}, \frac{2}{5})$, $(\bar{\phi}_1, \bar{\phi}_2) = (\frac{1}{3}, \frac{1}{3})$, and $(\bar{\phi}_1, \bar{\phi}_2) = (\frac{1}{4}, \frac{1}{4})$. Fig. 12 demonstrates that our method effectively separates an immiscible fluid mixture into islands of separate components as expected. Notably, the total mass $M_t = \int_\Omega \rho(\boldsymbol{x}, t) dV$ remains constant throughout the simulation, underscoring the accuracy of our solver.

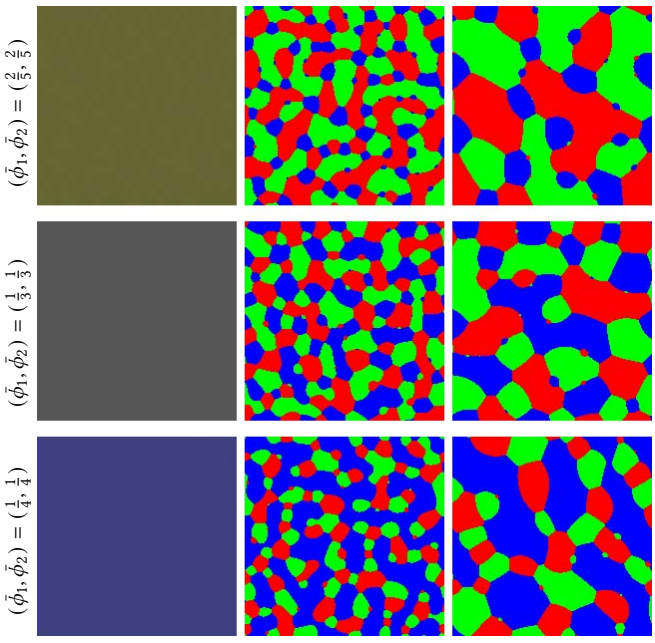


Fig. 12. **Spinodal decomposition evolution**: Each row shows the temporal evolution of a three-fluid system over time, for different initial fluid distributions as indicated on the left.

Raleigh-Taylor instability. The Raleigh-Taylor (RT) instability is the instability occurring at an interface between two fluids of different densities when the heavier fluid is pushing down on the lighter fluid [Cook et al. 2004; Abadi et al. 2018]. In our experiment, we employ three immiscible fluids with density ratios $\rho_3 : \rho_2 : \rho_1 = 3 : 2 : 1$, confined within a domain of size $L_0 \times 4L_0$ (where $L_0 = 256$). The heaviest fluid is on top, sharing a sinusoidal-shaped interface with the lighter fluid, and similarly between the lighter and lightest fluids. The phases are initialized as:

$$\begin{aligned}
\phi_1(\boldsymbol{x}, 0) &= 0.5 + 0.5 \tanh(2 * (y - 8 * L_0/3 + h(x)/W), \\
\phi_2(\boldsymbol{x}, 0) &= 0.5 + 0.5 \tanh(2 * (y - 4 * L_0/3 + h(x)/W) - \phi_1(\boldsymbol{x}, 0), \\
\phi_3(\boldsymbol{x}, 0) &= 1 - \phi_1(\boldsymbol{x}, 0) - \phi_2(\boldsymbol{x}, 0),
\end{aligned}$$

where $h(x) = 0.1 L_0 \sin(2\pi/L_0)$ represents the sinusoidal interface, $\xi = 5$ is the width of the interfaces, and the Reynolds number is $Re = \rho_1 L_0 U_0 / \nu_1 = 1000$. A dimensionless time t^* is defined as $t^* = t/\sqrt{L_0/gA_t}$, where $A_t = (\rho_1 - \rho_3)/(\rho_1 + \rho_3)$ and $g = 2 \times 10^{-5}$ denotes the downward gravitational acceleration. The characteristic velocity is $U_0 = \sqrt{gL_0}$. The simulation employs periodic boundaries on the left and right, while the upper and lower boundaries are handled using a half-way bounce-back scheme. Fig. 13 illustrates the temporal evolution of these three fluids, showing obvious similarities with the figure in [Abadi et al. 2018]. Initially, the heavy fluid goes down, creating a spike, while the light fluid ascends, forming a bubble. During the initial stages, viscous forces strengthen as the convective velocity between different fluids intensifies. This heightened viscosity increases resistance on both sides of the convex interface surrounded by the other fluids. Consequently, the movement of the interface on both sides decelerates compared to the middle part of the convex interface. This gradual slow-down results in the formation of a hook shape opposite to the direction of motion. As the interfaces continue to deform, the three-phase fluid system becomes unstable and undergoes mixing. After a while, the simulation experiences stratification again, forming a new three-layered structure with the lighter fluid on top.

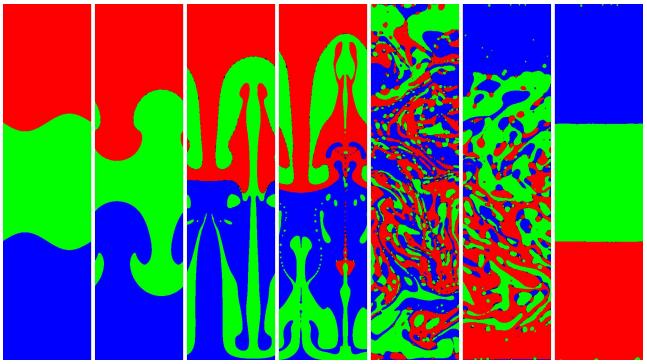


Fig. 13. **Raleigh-Taylor evolution**: From left to right, temporal evolution of a three-fluid RT instability (with side periodic condition).

To offer a quantitative comparison, we also calculate the average heights $\bar{Y}^i$ of the interface of the i^{th} fluid defined as:

$$\bar{Y}^i = \frac{\sum_\Omega y \phi(\boldsymbol{x}, t)}{\sum_\Omega \phi(\boldsymbol{x}, t)} \quad \text{for} \quad i = 1, 2, 3$$

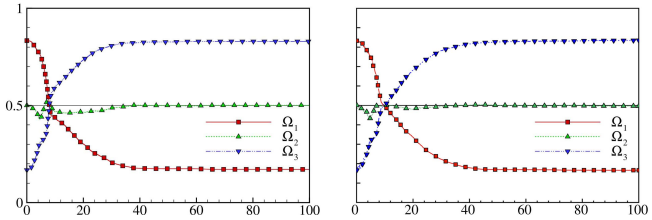


Image from [Abadi et al. 2018] *Ours*

Fig. 14. **Rayleigh-Taylor statistics**: Time evolution of the barycenter heights of each fluid, with red and blue denoting the heaviest and lightest fluid respectively.

Fig. 14 shows the averages $\{\bar{Y}^i\}$ in time, which are consistent with the results shown in [Abadi et al. 2018].

7.2 Comparisons with existing works

We also provide comparisons with recent CG results: the LBM-based framework of Li et al. [2022] and the multiphase Navier-Stokes work of Yan and Ren [2023].

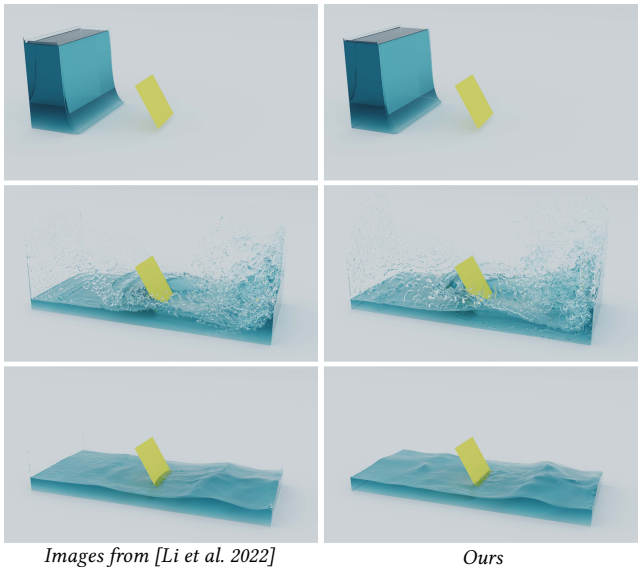
Comparison with [Li et al. 2022]. Our solver satisfies reduction consistency so that it can trivially simulate two-phase flows. A first test is thus simulated using our moment-based multiphase fluid solver alongside a distribution-based Lattice Boltzmann Method solver [Li et al. 2022], both with the same fluid and phase resolution. Leveraging our moment-based representation, the hybrid bounce-back treatment used in the distribution-based LBM solver can be eliminated, resulting in more reasonable bubbles without excessive pressure filtering, see Fig. 15,. More importantly, the distribution-based LBM solver requires 48 floats per node, while ours only needs 32 floats per node, for which 10 moment variables in the moment buffer can be reused to store temporal variables such as force, density, and $\Delta\phi$. This allows us to save 16 values per grid node in 3D, reducing the total memory usage by approximately 33%. Consequently, this reduced memory access allows for a significant speedup, as our solver ends up being approximately 2.35 times faster than the method proposed in [Li et al. 2022] at a resolution of $720 \times 360 \times 324$.

Comparison with [Yan and Ren 2023]. To compare with [Yan and Ren 2023], we run a three-phase dam break with density ratios $1500 : 30 : 1$ to match one of their examples, using a $280 \times 140 \times 140$ flow grid and $560 \times 280 \times 280$ phase grid as shown in Fig. 16. Similar to [Yan and Ren 2023], our solver maintains the property of phase-mass conservation. However, [Yan and Ren 2023] does not support the actual simulation of the air phase, and their simulation of the two other phases with 73K particles does not exhibit much detail. Comparatively, our true three-phase approach demonstrates very detailed splashing, bubbles, and wetting — obviously, at a computational cost around 50 times higher on this example.

7.3 Simulation results

Finally, we go over the various examples we ran with our solver to illustrate further the generality, efficiency, and realism of our n-phase solver.

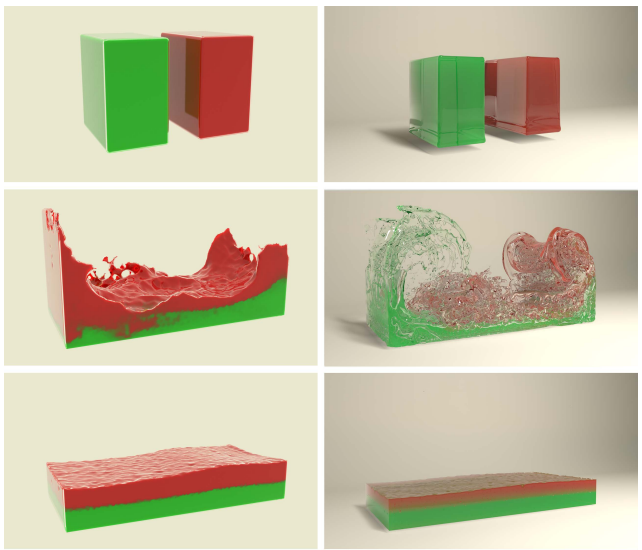
Dam breaks. We conducted dam break tests with three phases (two fluids and air) for density ratios $800 : 80 : 1$. In Fig. 2, a thin plate



Images from [Li et al. 2022] *Ours*

Fig. 15. **Comparison with [Li et al. 2022]**: For the same scene setup, our method (right) achieves 2.35 times speedup at a resolution of $720 \times 360 \times 324$, and does not show spurious ringing artifacts near solid boundaries.

with a 30-degree incline creates a pronounced crown splash as the water encounters the obstacle. The heavier phases sweep away the lighter phases, resulting in the formation of bubbles and splashes when the phases come in contact. Additionally, wetting patterns emerge as different density phases spatter onto the four transparent walls of the container. In this immiscible example, distinct layers of the different phases become clearly visible when the simulation reaches a steady state. Another dam break with, this time, a bunny-shaped obstacle, is shown in Fig. 9 to demonstrate our solver's capability in handling complex boundaries.



Images from [Yan and Ren 2023] *Ours*

Fig. 16. **Comparison with [Yan and Ren 2023]**: For a similar dam break setting, our method considers air a third phase, which rises to more splashing, the formation of bubbles, and wetting.

Table 3. **Statistics.** All examples run on a Nvidia 4090 GPU, with timings indicated for 60-fps sequences.

	Resolutions – Flow/Phase(s)	Viscosity ratio	Density ratio	Sec./frame
Fig. 1 & Fig. 17	162 ×270 ×162/324 ×540 ×324	0.00036 : 0.001 : 0.006	800 : 400 : 1	28.8
Fig. 2	480 ×240 ×216/960 ×480 ×432	0.00072 : 0.0024 : 0.006	800 : 40 : 1	143.7
Fig. 3	280 ×280 ×140/560 ×560 ×280	0.00084 : 0.0012 : 0.007	800 : 80 : 1	51.7
Fig. 4	400 ×200 ×180/800 ×400 ×360	0.0006 : 0.002 : 0.002 : 0.005	800 : 80 : 16 : 1	186.7
Fig. 5	120 ×240 ×120/240 ×480 ×240	0.00048 : 0.0012 : 0.0036	800 : 40 : 1	51.3
Fig. 7	336 ×168 ×336/672 ×336 ×672	0.0004 : 0.001 : 0.006	2000 : 200 : 1	105.5
Fig. 8	440 ×220 ×198/880 ×440 ×396	0.00066 : 0.00088 : 0.0055	800 : 80 : 1	111.2
Fig. 9	480 ×240 ×216/960 ×480 ×432	0.0006 : 0.0012 : 0.0024	800 : 40 : 1	164.4
Fig. 11	286 ×440 ×176/572 ×880 ×352	0.00044 : 0.0011 : 0.0011	800 : 720 : 1	140.2
Fig. 15 Ours	720 ×360 ×324/720 ×360 ×324	0.0009 : 0.072	800 : 1	18.8
Fig. 15 [Li et al. 2022]	720 ×360 ×324/720 ×360 ×324	0.0009 : 0.072	800 : 1	44.1
Fig. 16 Ours	280 ×140 ×140/560 ×280 ×280	0.00084 : 0.0012 : 0.0112	1500 : 30 : 1	13.4
Fig. 19 Top	144 ×240 ×144/288 ×480 ×288	0.00032 : 0.0008 : 0.008	1000 : 500 : 1	32.3
Fig. 19 Bottom	144 ×240 ×144/288 ×480 ×288	0.00768 : 0.0192 : 0.008	1000 : 500 : 1	32.3
Fig. 20	384 ×144 ×384/768 ×288 ×768	0.00064 : 0.00064 : 0.00096	1000 : 250 : 1	135.6
Fig. 21	440 ×209 ×198/880 ×418 ×396	0.00088 : 0.0022 : 0.0011	1000 : 700 : 1	108.6
Fig. 22	320 ×160 ×160/640 ×320 ×320	0.012 : 0.012 : 0.012 : 0.032	1000 : 100 : 25 : 1	32.5

Glugging. Our solver is capable of simulating a three-phase glugging effect, a phenomenon rarely showcased in previous works. The density ratios of the two fluids and the air are set as 800 : 400 : 1. We present two setups featuring different initial positions of the heaviest phase. In Fig. 1, the dense fluid (blue) is placed on top of the light fluid (green), in the upper part of an hourglass container. As the simulation begins, a Rayleigh–Taylor instability appears, followed by intricate bubbling, guggling, and wetting. Fig. 17 shows another setup where the dense fluid is positioned below the light fluid. The unified nature of our solver targeting a spectrum of multiphase phenomena can exhibit a wide range of bubble sizes deep within the liquid depending on the initial conditions. In both cases, at the end of the simulations, distinct layers of the two different fluids are clearly visible, and the respective volume occupancy of each fluid is conserved.

Unmixing a mixture. In Fig. 3, an immiscible mixture of three phases (more precisely two fluids and air) is initialized with equal phase values at each fluid node (amounting to a preliminary vigorous shake of the container), and evolves by eventually separating into different layered parts due to the different density ratios set to 800 : 80 : 1. The animation reveals natural settling of the fluids under gravity, capturing multiple bubbles. Due to wetting and surface tension effects, some small drops are even left on the top wall of the container. The equal division at the beginning state ensures that our solver generates an exact equal phase separation at the end of the animation. To showcase more complex splashing, we also conducted a test involving a mixture of gray droplets onto the floor, resulting in splashing and bubble generation before the fluids separate into red and blue phases, as seen in Fig. 8.

Porous media. To test robustness, we simulate fluids interacting with highly complex obstacles. Fig. 5 illustrates two immiscible fluids (plus air) traversing a porous rock with density ratios 800 : 80 : 1. Initially, the dense blue fluid displaces the lighter green fluid, after which the blue liquid navigates through the irregular holes and tunnels first, with the green liquid following closely behind. During this process, bubbles form, and liquid filaments and drops emerge at the bottom of the rock as water exits. In Fig. 19, two different immiscible liquids (plus air) fall down an S-shaped container, making their way through two separating plates with hydrophobic wetting boundary conditions. Here, the density ratios are set to 1000 : 500 : 1.

Immiscible/Miscible multiphase flows. We further simulate a three-fluid dam break, where the three fluids are immiscible with air, for density ratios 800 : 80 : 16 : 1. Fig. 4 demonstrates three different miscibility coefficients between the three fluids. The first row illustrates a scenario where the three fluids are miscible with each other, and as time progresses, they indeed merge into a single fluid. The second row depicts a case where the three phases are partially miscible, and a blurred interface between the fluids becomes visible. The last row presents a scenario where the three phases are immiscible, forming clear interface layers at the end of the simulation.

Water into oil. Fig. 11 shows a ball of water dropping into an oil container whose density ratio is set 800 : 720 : 1, close to real-life ratios, creating a water-in-oil emulsion.

Various boundary conditions. To showcase different boundary conditions, we also provide inlet and outlet flow scenario and a one-way coupling simulation: in Fig. 20, a fan (with a constant rotation speed at the beginning before gradually shutting down) creates vorticity in the air phase and drives the motion of the other two flows; in Fig. 21, an inlet continuously blows air onto a two-fluid mixture, creating splashes and mixing.

Surface tension effects. In Fig. 22, we test surface tension effects by showing how a cube-shaped drop evolves into a spherical shape faster or slower depending on the density of the fluid: the lightest fluid (blue) converges to the final state more quickly than the others.

Large density ratios. Finally, to demonstrate our solver's ability to handle very large density ratios, we ran a dam break with two different fluids and air, for density ratios 2000 : 200 : 1, see Fig. 7.

8 CONCLUSION

In this paper, we have presented a new kinetic solver to simulate multifluid flows, where each of the fluids or gas can have its own density, miscibility, viscosity, and capillarity. Besides producing far more complex fluid animations, the massively parallel nature of our solver results in extremely efficient computations compared to previous works for comparable visual complexity. We designed this general fluid simulator by contributing a number of elements in the usual time integration pipeline of LBM methods:

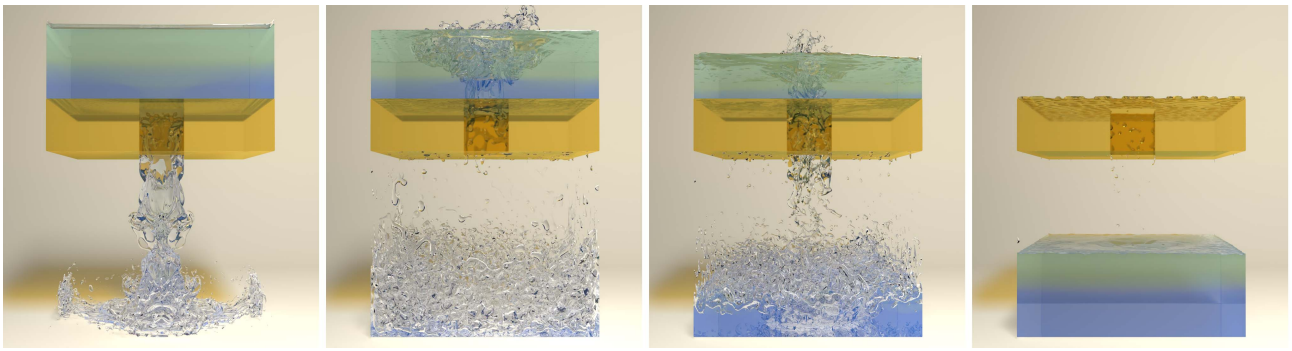


Fig. 17. **Three-phase glugging.** As two liquids of different densities are flowing down an hour-glass like container, the heavier flow (blue) hits the ground first and forms bubbles going across the second fluid inside the top container. Splashes and bubbles are also observed in the lower container. Near the end, wetting is seen along the container's walls, and the liquids form two different layers again.

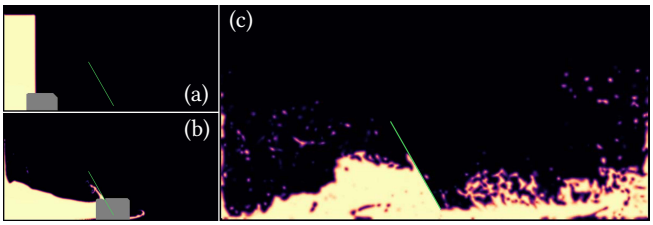


Fig. 18. **Ablation study.** For a 3D simulation scenario similar to Fig. 15 and visualized as a cut-slice in the phase field, using a momentum-based LBM distribution [Fakhari et al. 2017a] for the flow crashes early on because of the large density ratio (2000) (a), while a velocity-based distribution [Fakhari et al. 2017b] only fails when the bounce-back treatment of the fluid hitting the obstacle kicks in (b); only our velocity-based approach encoded via three moments maintains stability throughout the simulation (c).

- We adapted the D3Q27(or D3Q19) HOME-LBM method of Li et al. [2023] to our case of multiphase flows by providing new closed-form third-order-accurate expressions of both the *filtered reconstruction* of a "velocity-based" distribution from its *three first velocity moments* (Eq. (28)) and the effects of a central-moment MRT collision model on these three velocity moments (Eq. (29), (30) and (31)). These expressions not only reduce memory requirements (the main Achilles heel of kinetic solvers) by two-thirds but also greatly accelerate the kinetic integration of the velocity field by avoiding the costly projections between distributions and central moments that are used in state-of-the-art collision models.
- Our phase-field D3Q7 representation of the spatial occupancy of the fluids and gas is advanced in time by a conservative time update equation given in Eq. (12) which we extended from the CPF method [Fakhari et al. 2017b] through a constrained minimization to account for the preservation of respective fluid volumes, offering phase reduction consistency as well.
- The kinetic solver for the velocity field can thus pass the current velocity to advect phases, while the phase solver returns interfacial forces based on phases and densities to affect the evolution of the velocity field.

These contributions result in an efficient and stable multiphase integrator, which can handle high Reynolds numbers and large density ratios. Given our adjustable fluid characteristics like miscibility and wetting boundary conditions, we were able to show a variety of complex simulations to demonstrate the generality and efficiency of our work in generic simulation cases.

Limitations and future work. While we use a finer phase grid than the velocity grid to improve the phases' ability to perform a more accurate evolution and to accommodate both thick and thin obstacles, our diffuse interface model can still lose small details at scales smaller than the grid, which would require an adaptive grid approach to fully resolve. We leave this as future work, although the approach of Lyu et al. [2023], once adapted to our multiphase context, might be a good way to introduce adaptivity. An obvious next step would be to use HOME-LBM for the phases as well, to avoid having to compute the collision operator on the distributions. However, our current use of D3Q7 lattice is not particularly memory or time consuming, so we may not gain much by storing the first two velocity-moments instead — although in-place streaming and collision-related computations are the bottleneck in our multifluid solver. Moreover, our current solver has not been tested for two-way coupling: while the approach of Li and Desbrun [2023] can be applied to our solver to achieve this, we have not been able to test it thoroughly enough, and chances are that modifications will be required to offer the same expected stability; we thus provided mostly scenes with fixed boundaries in this paper and only one-way examples. Finally, the current phase field model does not support phase changes. Introducing a pseudopotential model to deal with liquid–vapor phase change [Fei et al. 2020] into our multiphase simulator could bring a whole slew of interesting applications, such as the distillation of a mixture.

ACKNOWLEDGMENTS

We thank the reviewers for their help and our colleagues from LightSpeed Studios, Fengquan Wang and Dong Li, for their support of the project. The bunny model in Fig. 9 is from Stanford. MD acknowledges the generous support of Ansys, Adobe Research, and the MediTwin consortium, as well as a Choose France Inria chair.

REFERENCES

Reza Haghani Hassan Abadi, Mohammad Hassan Rahimian, and Abbas Fakhari. 2018. Conservative phase-field lattice-Boltzmann model for ternary fluids. *J. Comput. Phys.* 374 (2018), 668–691.

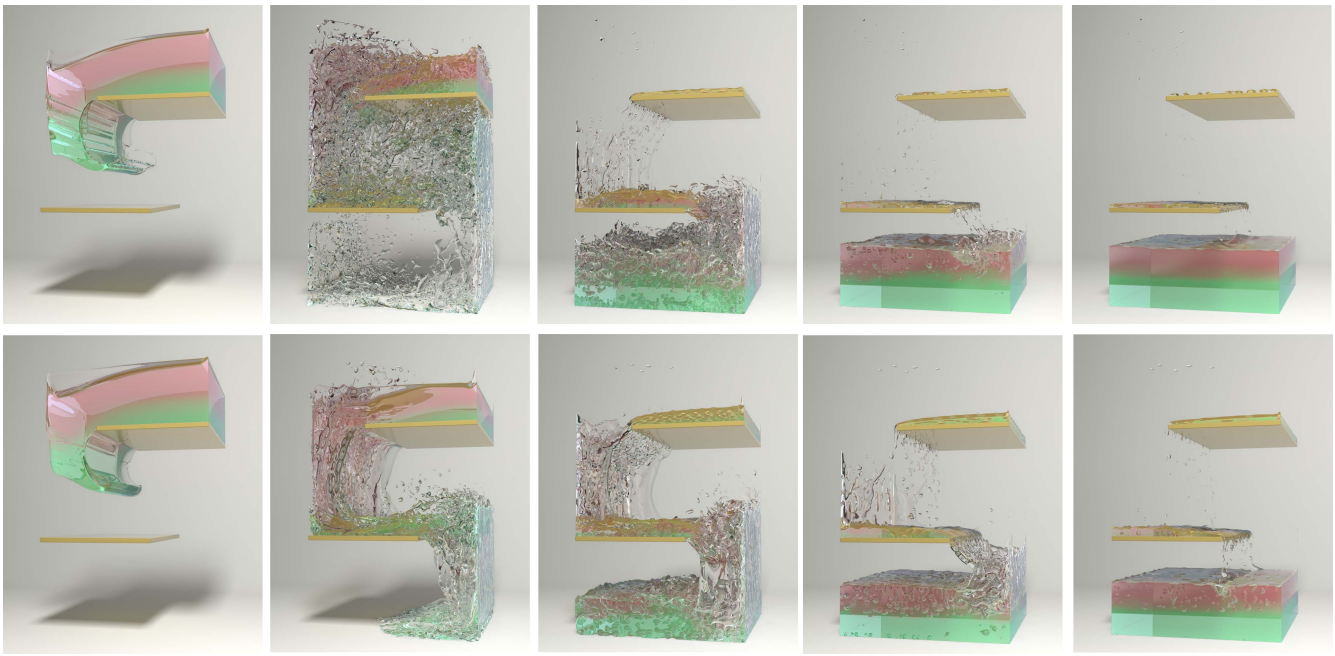


Fig. 19. **Multiphase flow in S-shape container.** This example shows two liquids flowing down over two separating hydrophobic plates, forming splashes, bubbles, and wetting effects (top). Increasing the viscosity of each phase results in a smoother, slower motion (bottom).

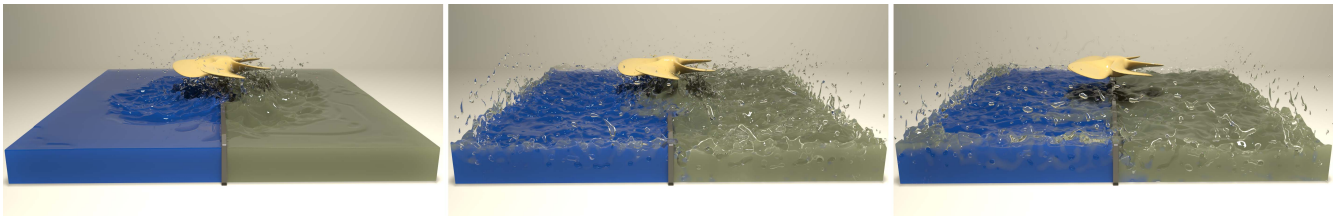


Fig. 20. **Spinning fan creates eddies.** A fan (with a constant rotation speed at the beginning before gradually coming to rest) creates vorticity in the air phase and drives the motion of the other two flows which are separated by a small partition.

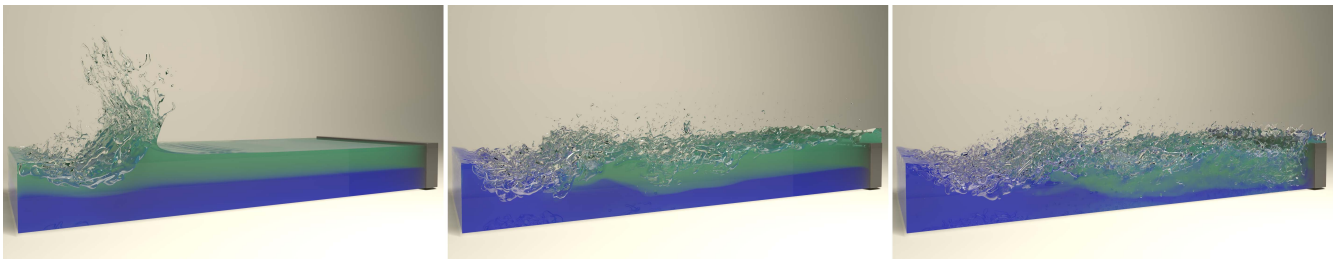


Fig. 21. **Air-driven motion of multifluids.** An inlet in air phase continuously blows on the two fluids and generate a mixture and splashes.

Samuel M Allen and John W Cahn. 1976. Mechanisms of phase transformations within the miscibility gap of Fe-rich Fe-Al alloys. *Acta Metallurgica* 24, 5 (1976), 425–437.

Daniel M Anderson, Geoffrey B McFadden, and Adam A Wheeler. 1998. Diffuse-interface methods in fluid mechanics. *Annual review of fluid mechanics* 30, 1 (1998), 139–165.

Ryoichi Ando, Nils Thuerey, and Chris Wojtan. 2015. A Stream Function Solver for Liquid Simulations. *ACM Trans. Graph.* 34, 4, Article 53 (jul 2015), 9 pages.

Kai Bao, Xiaolong Wu, Hui Zhang, and Enhua Wu. 2010. Volume Fraction Based Miscible and Immiscible Fluid Animation. *Comput. Animat. Virtual Worlds* 21, 3-4 (may 2010), 401–410.

Weizhu Bao and Qiang Du. 2011. *Multiscale Modeling and Analysis for Materials Simulation.* World Scientific, Singapore.

Landon Boyd and Robert Bridson. 2012. MultiFLIP for Energetic Two-Phase Fluid Simulation. *ACM Trans. Graph.* 31, 2, Article 16 (apr 2012), 12 pages.

Morgan Brassel and Elie Bretin. 2011. A modified phase field approximation for mean curvature flow with conservation of the volume. *Mathematical Methods in the Applied Sciences* 10, 34 (2011), 1157–1180.

John W Cahn and John E Hilliard. 1958. Free energy of a nonuniform system. I. Interfacial free energy. *The Journal of chemical physics* 28, 2 (1958), 258–267.

Zhenhua Chai, Dongke Sun, Huili Wang, and Baochang Shi. 2018. A comparative study of local and nonlocal Allen-Cahn equations with mass conservation. *International Journal of Heat and Mass Transfer* 122 (2018), 631–642.

Yixin Chen, Wei Li, Rui Fan, and Xiaopei Liu. 2022. GPU Optimization for High-Quality Kinetic Fluid Simulation. *IEEE Trans. Vis. Comp. Graph.* 28, 9 (2022), 3235–3251.

Fig. 22. **Cubic droplet deformation.** A cube-shaped drop evolves into a spherical shape faster or slower depending on the density of the fluid (the density ratio of green:red:blue:air phase is 1000 : 100 : 25 : 1): the lightest fluid (blue) converges to the final state much more quickly than the others.

Pao-Hsiung Chiu and Yan-Ting Lin. 2011. A conservative phase field method for solving incompressible two-phase flows. *J. Comput. Phys.* 230, 1 (2011), 185–204.

Andrew W Cook, William Cabot, and Paul L Miller. 2004. The mixing transition in Rayleigh–Taylor instability. *Journal of Fluid Mechanics* 511 (2004), 333–362.

Mathieu Desbrun and Marie-Paule Gascuel. 1996. Smoothed Particles: A New Paradigm for Animating Highly Deformable Bodies. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation '96* (Poitiers, France). Springer-Verlag, Berlin, Heidelberg, 61–76.

Dominique d'Humières. 2002. Multiple–relaxation–time lattice Boltzmann models in three dimensions. *Phil. Trans. R. Soc. A* 360 (2002), 437–451. Issue 1792.

Hang Ding, Peter DM Spelt, and Chang Shu. 2007. Diffuse interface model for incompressible two-phase flows with large density ratios. *J. Comput. Phys.* 226, 2 (2007), 2078–2095.

Suchuan Dong. 2017. Wall-bounded multiphase flows of N immiscible incompressible fluids: Consistency and contact-angle boundary condition. *J. Comput. Phys.* 338 (2017), 21–67.

Charles M Elliott and Stefan Luckhaus. 1991. A generalised diffusion equation for phase separation of a multi-component mixture with interfacial free energy. *Retrieved from the University of Minnesota Digital Conservancy* 1 (1991), 29 pages.

Abbas Fakhari, Diogo Bolster, and Li-Shi Luo. 2017a. A weighted multiple-relaxation-time lattice Boltzmann method for multiphase flows and its application to partial coalescence cascades. *J. Comput. Phys.* 341 (2017), 22–43.

Abbas Fakhari, Travis Mitchell, Christopher Leonardi, and Diogo Bolster. 2017b. Improved locality of the phase-field lattice-Boltzmann model for immiscible fluids at high density ratios. *Physical Review E* 96, 5 (2017), 053301.

Linlin Fei, Jiapei Yang, Yiran Chen, Huangrui Mo, and Kai H Luo. 2020. Mesoscopic simulation of three-dimensional pool boiling based on a phase-change cascaded lattice Boltzmann method. *Physics of Fluids* 32, 10 (2020).

Ming Gao, Andre Pradhana, Xuchen Han, Qi Guo, Grant Kot, Eftychios Sifakis, and Chenfanfu Jiang. 2018a. Animating fluid sediment mixture in particle-laden flows. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–11.

Ming Gao, Xinlei Wang, Kui Wu, Andre Pradhana, Eftychios Sifakis, Cem Yuksel, and Chenfanfu Jiang. 2018b. GPU Optimization of Material Point Methods. *ACM Trans. Graph.* 37, 6, Article 254 (dec 2018), 12 pages.

Qiang He, Yongjian Li, Weifeng Huang, Yang Hu, Decai Li, and Yuming Wang. 2020. A unified lattice Boltzmann model for immiscible and miscible ternary fluids. *Computers & Mathematics with Applications* 80, 12 (2020), 2830–2859.

Jeong-Mo Hong and Chang-Hun Kim. 2005. Discontinuous Fluids. *ACM Trans. Graph.* 24, 3 (jul 2005), 915–920.

Yang Hu, Decai Li, and Qiang He. 2020. Generalized conservative phase field model and its lattice Boltzmann scheme for multicomponent multiphase flows. *International Journal of Multiphase Flow* 132 (2020), 103432.

Y. Jiang and Y. Lan. 2021. A Dynamic Mixture Model for Non-equilibrium Multiphase Fluids. *Computer Graphics Forum* 40, 7 (2021), 85–95.

Y. Jiang, C. Li, S. Deng, and S. M. Hu. 2020. A Divergence-free Mixture Model for Multiphase Fluids. *Computer Graphics Forum* 39, 8 (2020), 69–77.

Nahyup Kang, Jinho Park, Junyong Noh, and Sung Yong Shin. 2010. A Hybrid Approach to Multiple Fluid Simulation using Volume Fractions. *Computer Graphics Forum* 29, 2 (2010), 685–694.

Byungmoon Kim. 2010. Multi-phase fluid simulations using regional level sets. *ACM Transactions on Graphics (TOG)* 29, 6 (2010), 1–8.

Junseok Kim and Hyun Geun Lee. 2017. A new conservative vector-valued Allen–Cahn equation and its fast numerical method. *Computer Physics Communications* 221 (2017), 102–108.

Hyun Geun Lee and Junseok Kim. 2012. An efficient and accurate numerical algorithm for the vector-valued Allen–Cahn equations. *Computer Physics Communications* 183, 10 (2012), 2107–2115.

Wei Li, Yixin Chen, Mathieu Desbrun, Changxi Zheng, and Xiaopei Liu. 2020. Fast and Scalable Turbulent Flow Simulation with Two-Way Coupling. *ACM Trans. Graph.* 39, 4, Article 47 (2020).

Wei Li and Mathieu Desbrun. 2023. Fluid-Solid Coupling in Kinetic Two-Phase Flow Simulation. *ACM Trans. Graph.* 42, 4, Article 123 (jul 2023), 14 pages.

Wei Li, Daoming Liu, Mathieu Desbrun, Jin Huang, and Xiaopei Liu. 2021. Kinetic-Based Multiphase Flow Simulation. *IEEE Trans. Vis. Comp. Graph.* 27, 7 (2021), 3318–3334.

Wei Li, Yihui Ma, Xiaopei Liu, and Mathieu Desbrun. 2022. Efficient Kinetic Simulation of Two-Phase Flows. *ACM Trans. Graph.* 41, 4, Article 114 (jul 2022), 17 pages.

Wei Li, Tongtong Wang, Zherong Pan, Xifeng Gao, Kui Wu, and Mathieu Desbrun. 2023. High-Order Moment-Encoded Kinetic Simulation of Turbulent Flows. *ACM Trans. Graph.* 42, 6 (2023), 1–13.

Frank Losasso, Tamar Shinar, Andrew Selle, and Ronald Fedkiw. 2006. Multiple interacting liquids. *ACM Transactions on Graphics (TOG)* 25, 3 (2006), 812–819.

Qin Lou, Zhaoli Guo, and Baochang Shi. 2013. Evaluation of outflow boundary conditions for two-phase lattice Boltzmann equation. *Physical review E* 87, 6 (2013), 063301.

Chaoyang Lyu, Kai Bai, Yiheng Wu, Mathieu Desbrun, Changxi Zheng, and Xiaopei Liu. 2023. Building a Virtual Weakly-Compressible Wind Tunnel Testing Facility. *ACM Trans. Graph.* 42, 4 (2023).

V. Mihalef, D. Metaxas, and M. Sussman. 2007. Textured Liquids based on the Marker Level Set. *Computer Graphics Forum* 26, 3 (2007), 457–466.

Matthias Müller, Barbara Solenthaler, Richard Keiser, and Markus Gross. 2005. Particle-Based Fluid-Fluid Interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Los Angeles, California) *(SCA '05)*. Association for Computing Machinery, New York, NY, USA, 237–244.

Michael B. Nielsen and Ole Østerby. 2013. A Two-Continua Approach to Eulerian Simulation of Water Spray. *ACM Trans. Graph.* 32, 4, Article 67 (jul 2013), 10 pages.

Stanley Osher and James A Sethian. 1988. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of computational physics* 79, 1 (1988), 12–49.

Simon Premžoe, Tolga Tasdizen, James Bigler, Aaron Lefohn, and Ross T. Whitaker. 2003. Particle-Based Simulation of Fluids. *Computer Graphics Forum* 22, 3 (2003), 401–410.

Bo Ren, Wei He, Chen-Feng Li, and Xu Chen. 2021. Incompressibility Enforcement for Multiple-Fluid SPH Using Deformation Gradient. *IEEE Transactions on Visualization and Computer Graphics* 28, 10 (2021), 3417–3427.

Bo Ren, Chenfeng Li, Xiao Yan, Ming C Lin, Javier Bonet, and Shi-Min Hu. 2014. Multiple-fluid SPH simulation using a mixture model. *ACM Transactions on Graphics (TOG)* 33, 5 (2014), 1–11.

Bo Ren, Xu-Yun Yang, Ming C. Lin, Nils Thuerey, Matthias Teschner, and Chenfeng Li. 2018. Visual Simulation of Multiple Fluids in Computer Graphics: A State-of-the-Art Report. *Journal of Computer Science and Technology* 33, 3 (2018), 431–451.

Jacob Rubinstein and Peter Sternberg. 1992. Nonlocal reaction—diffusion equations and nucleation. *IMA Journal of Applied Mathematics* 48, 3 (1992), 249–264.

Xiaowen Shan, Xue-Feng Yuan, and Hudong Chen. 2006. Kinetic theory representation of hydrodynamics: a way beyond the Navier–Stokes equation. *J. Fluid Mech.* 550 (2006), 413–441.

Jos Stam. 1999. Stable Fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley

Publishing Co., USA, 121–128.
Ying Sun and Christoph Beckermann. 2007. Sharp interface tracking using the phase-field equation. *J. Comput. Phys.* 220, 2 (2007), 626–653.
Mark Sussman, Peter Smereka, and Stanley Osher. 1994. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational physics* 114, 1 (1994), 146–159.
Grétar Tryggvason, Bernard Bunner, Asghar Esmaeeli, Damir Juric, N Al-Rawahi, W Tauber, J Han, S Nas, and Y-J Jan. 2001. A front-tracking method for the computations of multiphase flow. *Journal of computational physics* 169, 2 (2001), 708–759.
Huili Wang, Xiaolei Yuan, Hong Liang, Zhenhua Chai, and Baochang Shi. 2019. A brief review of the phase-field-based lattice Boltzmann method for multiphase flows. *Capillarity* 2, 3 (2019), 33–52.
Wolfram Research Inc. 2023. Mathematica, Version 13.2. https://www.wolfram.com/mathematica
Shuonan Wu and Jinchao Xu. 2017. Multiphase Allen–Cahn and Cahn–Hilliard models and their discretizations with the effect of pairwise surface tensions. *J. Comput. Phys.* 343 (2017), 10–32.
Han Yan and Bo Ren. 2023. High Density Ratio Multi-Fluid Simulation with Peridynamics. *ACM Trans. Graph.* 42, 6, Article 191 (2023), 14 pages.
Xiao Yan, Yun-Tao Jiang, Chen-Feng Li, Ralph R Martin, and Shi-Min Hu. 2016. Multiphase SPH simulation for interactive fluids and solids. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–11.
Xiao Yan, Chen-Feng Li, Xiaosong Chen, and Shi-Min Hu. 2018. MPM simulation of interacting fluids and solids. *Computer Graphics Forum* 37, 8 (2018), 183–193.
Tao Yang, Jian Chang, Ming C Lin, Ralph R Martin, Jian J Zhang, and Shi-Min Hu. 2017. A unified particle system framework for multi-phase, multi-material visual simulations. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–13.
Tao Yang, Jian Chang, Bo Ren, Ming C Lin, Jian Jun Zhang, and Shi-Min Hu. 2015. Fast multiple-fluid simulation using Helmholtz free energy. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 1–11.
Pengtao Yue, Chunfeng Zhou, and James J Feng. 2007. Spontaneous shrinkage of drops and mass conservation in phase-field simulations. *J. Comput. Phys.* 223, 1 (2007), 1–9.
Lin Zheng and Song Zheng. 2019. Phase-field-theory-based lattice Boltzmann equation method for N immiscible incompressible fluids. *Physical Review E* 99, 6 (2019), 063310.
Hongbin Zhu, Kai Bao, Enhua Wu, and Xuehui Liu. 2007. Stable and Efficient Miscible Liquid-Liquid Interactions. In *Proceedings of the 2007 ACM Symposium on Virtual Reality Software and Technology* (Newport Beach, California) *(VRST '07)*. Association for Computing Machinery, New York, NY, USA, 55–64.
Hongbin Zhu, Xuehui Liu, Youquan Liu, and Enhua Wu. 2006. Simulation of miscible binary mixtures based on lattice Boltzmann method. *Computer Animation and Virtual Worlds* 17, 3-4 (2006), 403–410.

A HERMITE POLYNOMIALS

The explicit second-order Hermite tensor values we need are:

$$H^{[2]}_{\alpha,\beta}(\boldsymbol{c}_i) = c_{i,\alpha}c_{i,\beta} - \tfrac{1}{3}\delta_{\alpha\beta},$$

while the third-order Hermite tensor values are:

$$H^{[3]}_{\alpha,\beta,\gamma}(\boldsymbol{c}_i) = c_{i,\alpha}c_{i,\beta}c_{i,\gamma} - \tfrac{1}{3}\left(c_{i,\alpha}\delta_{\beta\gamma} + c_{i,\beta}\delta_{\alpha\gamma} + c_{i,\gamma}\delta_{\alpha\beta}\right).$$

B 2D MOMENT-BASED RECONSTRUCTION

The 2D reconstruction of a distribution from its three velocity moments p^*, $\boldsymbol{u}$, and $\boldsymbol{S}$ is also provided, in order to complement the 3D expression from Eq. (28):

$$\begin{aligned} g_i =& w_i\Bigg[p^* + \frac{\boldsymbol{c}_i\cdot\boldsymbol{u}}{c_s^2} + \frac{\boldsymbol{H}^{[2]}(\boldsymbol{c}_i):\mathbf{S}}{2c_s^4} \\ &+ \frac{1}{2c_s^6}\big(H^{[3]}_{xxy}(\boldsymbol{c}_i)(S_{xx}u_y + 2S_{xy}u_x - 2u_xu_xu_y) \\ &\quad + H^{[3]}_{xyy}(\boldsymbol{c}_i)(S_{yy}u_x + 2S_{xy}u_y - 2u_xu_yu_y)\big)\Bigg]. \end{aligned} \tag{40}$$

C 2D CENTRAL-MOMENT-BASED COLLISION

In 2D, the moment-based collision's update equations become:

$$\begin{aligned} p^*(\boldsymbol{x}, t+1) &= p^{*'}; \\ u_\alpha(\boldsymbol{x}, t+1) &= u^*_\alpha + \tfrac{1}{2\rho'}F_\alpha; \\ S_{xy}(\boldsymbol{x}, t+1) &= (1-\tfrac{1}{\tau})S^*_{xy} + \tfrac{1}{\tau}u^*_xu^*_y + \tfrac{2\tau-1}{2\tau\rho'}(F_xu^*_y + F_yu^*_x); \\ S_{xx}(\boldsymbol{x}, t+1) &= \tfrac{\tau-1}{2\tau}(S^*_{xx} - S^*_{yy}) + \tfrac{\tau+1}{2\tau}u^{*2}_x \\ &\quad + \tfrac{\tau-1}{2\tau}u^{*2}_y + \tfrac{1}{\rho'}F_xu^*_x + \tfrac{\tau-1}{2\tau\rho'}(F_xu^*_x - F_yu^*_y); \\ S_{yy}(\boldsymbol{x}, t+1) &= \tfrac{\tau-1}{2\tau}(S^*_{yy} - S^*_{xx}) + \tfrac{\tau+1}{2\tau}u^{*2}_y \\ &\quad + \tfrac{\tau-1}{2\tau}u^{*2}_x + \tfrac{1}{\rho'}F_yu^t_y + \tfrac{\tau-1}{2\tau\rho'}(F_yu^*_y - F_xu^*_x). \end{aligned}$$

D D3Q19 MOMENT-BASED RECONSTRUCTION

The moment-based distribution function reconstruction for the D3Q19 lattice structure (Fig. 6) ignores the cube diagonals in Eq. (28), and becomes:

$$\begin{aligned} g_i =& w_i\Bigg[p^* + \frac{\boldsymbol{c}_i\cdot\boldsymbol{u}}{c_s^2} + \frac{\boldsymbol{H}^{[2]}(\boldsymbol{c}_i):\boldsymbol{S}}{2c_s^4} \\ &+ \tfrac{1}{2c_s^6}\big(H^{[3]}_{xxy}(\boldsymbol{c}_i)(S_{xx}u_y + 2S_{xy}u_x - 2u_xu_xu_y) \\ &\quad + H^{[3]}_{xyy}(\boldsymbol{c}_i)(S_{yy}u_x + 2S_{xy}u_y - 2u_xu_yu_y) \\ &\quad + H^{[3]}_{xxz}(\boldsymbol{c}_i)(S_{xx}u_z + 2S_{xz}u_x - 2u_xu_xu_z) \\ &\quad + H^{[3]}_{xzz}(\boldsymbol{c}_i)(S_{zz}u_x + 2S_{xz}u_z - 2u_xu_zu_z) \\ &\quad + H^{[3]}_{yzz}(\boldsymbol{c}_i)(S_{zz}u_y + 2S_{yz}u_z - 2u_yu_zu_z) \\ &\quad + H^{[3]}_{yyz}(\boldsymbol{c}_i)(S_{yy}u_z + 2S_{yz}u_z - 2u_yu_yu_z)\big)\Bigg]. \end{aligned} \tag{41}$$

The moment-collision for $D3Q19$ lattice structure stays the same as Eqs. (29), (30) and (31).