A Unified Detail-Preserving Liquid Simulation by Two-Phase Lattice Boltzmann Modeling

Yulong Guo, Xiaopei Liu, and Xuemiao Xu

Abstract—Traditional methods in graphics to simulate liquid-air dynamics under different scenarios usually employ separate approaches with sophisticated interface tracking/reconstruction techniques. In this paper, we propose a novel unified approach which is easy and effective to produce a variety of liquid-air interface phenomena. These phenomena, such as complex surface splashes, bubble interactions, as well as surface tension effects, can co-exist in one single simulation, and are created within the same computational framework. Such a framework is unique in that it is free from any complicated interface tracking/reconstruction procedures. Our approach is developed from the two-phase lattice Boltzmann method with the mean field model, which provides a unified framework for interface dynamics but is numerically unstable under turbulent conditions. Considering the drawbacks of the existing approaches, we propose techniques to suppress oscillations for significant stability enhancement, as well as derive a new subgrid-scale model to further improve stability, faithfully preserving liquid-air interface details without excessive diffusion by taking into account the density variation. The whole framework is highly parallel, enabling very efficient implementation. Comparisons with the related approaches show superiority on stable simulations with detail preservation and multiphase phenomena simultaneously involved. A set of animation results demonstrate the effectiveness of our method.

Index Terms—Flow simulation, two-phase lattice Boltzmann method, interface flow

1 Introduction

The interesting dynamics of liquid-air interaction has inspired massive research on simulation techniques in computer graphics to produce visually realistic flow animations [1], [2]. Traditional methods usually model liquid-air interactions in different scenarios separately. For example, to simulate liquid-air surface splashes, a free-surface model is often adopted, with tracking techniques to trace the interface motion [3], [4], or reconstruction techniques to recover the interface in Lagrangian methods [5], [6]. However, the basic free surface model relies on additional techniques to produce multiphase phenomena, such as bubble dynamics and surface tension effects [7], [8], [9], by including the interaction between the liquid and gas phases. Moreover, in situations such as turbulent flows, the interface geometry becomes extremely complicated, which demands sophisticated solvers to retain high enough accuracy at the interface in order to obtain the desired results [10], [11].

Unlike traditional approaches, we propose in this paper a unified framework that is easy yet effective to solve for complex liquid-air interface dynamics, which simultaneously includes phenomena ranging from interface splashes to bubbles and surface tension flows. More importantly, our framework unifies the handling of different phases and incorporates inter-molecular forces for different kinds of phase interactions. The interface is automatically captured by an index function without complicated tracking/reconstruction procedures, which greatly reduces the complexity for simulation.

However, the existing TP-LBMs in computational physics [12], [13], [14] are difficult to be directly applied in computer graphics to simulate complex liquid-air interface phenomena, since it is hard to maintain sufficient stability in a turbulent environment, which is very common in many real scenarios. Therefore, the key to the success of our method is to stabilize TP-LBM in highly fluctuating turbulent circumstances. In this regard, we develop our framework from the TP-LBM with the mean-field model, and propose the following two important aspects as our technical contributions:

- First, we found that the oscillation is the main source of instability in the TP-LBM with the mean-field model when the flow becomes turbulent. There are several aspects which cause the oscillations such as the weak compressibility of the model where the compression waves can be generated, as well as the unified treatment of both phases. Thus, we propose a set of new techniques to suppress unwanted oscillations and thus overshooting, which significantly improves the model stability.
- Second, even with oscillations suppressed, it is still insufficient to simulate turbulent liquid-air interface flows, as the underlying equations become unstable again in such circumstances, and the interface details could not be faithfully preserved either. Thus, we

- *Y. Guo and X. Liu are with the School of Information Science and Technology, ShanghaiTech University, Shanghai, China. E-mail: {guoyl, liuxp}@shanghaitech.edu.cn.*
- *X. Xu is with the School of Computer Science and Engineering, South China University of Technology, Guangdong, China. E-mail: xuemx@scut.edu.cn.*

Manuscript received 20 Oct. 2015; revised 29 Jan. 2016; accepted 3 Feb. 2016. Date of publication 19 Feb. 2016; date of current version 29 Mar. 2017. Recommended for acceptance by J. Lee.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2016.2532335

derive a new subgrid-scale model to address this issue by taking into account the density variation and combine this subgrid-scale model with the oscillation-suppressed TP-LBM to form our final model.

The proposed computational framework is easy for parallel implementation due to the explicit formulation and the local updating rules. To demonstrate the effectiveness of our method, we simulated various liquid-air interface phenomena in different scenarios including liquid-air free surface flows with turbulent splashes, bubble flows with complex interactions, and surface-tension-driven flows. It is important to note that our method is unique in that it captures these phenomena without the need to explicitly identify the interface during the simulation, which is significantly different from the traditional approaches where interface identification or components coupling is required.

2 Related Work

The demand of fluid simulation in computer graphics over the past decade has inspired a large number of research with innovative techniques [5], [15], [16]. Here, we only review works that are closely related to the simulation of liquid-air interface flow phenomena.

Free surface liquid simulation. For grid-based Eulerian methods, Foster and Fedkiw [3] first introduced the level-set method to track the liquid surface. Enright et al. [4] proposed the particle level-set method to improve accuracy. To further increase interface details, Losasso et al. [17] took an adaptive approach by using octree. Heo and Ko [10] proposed to use sub-grid quadrature points. Bojsen-Hansen and Wojtan [11] used error compensation methods to reduce the artifacts for a high resolution tracker.

Particle-based methods, such as the smoothed particle hydrodynamics (SPH) [18], and some other hybrid methods, such as the Fluid-Implicit-Particle (FLIP) method [5] are also competitive in producing interface details. The improvements include better incompressibility for SPH [19], the adaptivity for FLIP discretization [20], and the introduction of position-based dynamics (PBD) [21], etc. However, the surface reconstruction with high quality requires complicated procedures in these methods, such as the anisotropic kernel approach of Yu and Turk [6].

In addition to the above methods, Thuerey and Ruede [22] combined the lattice Boltzmann method with the volume-of-fluid-style interface tracking for free surface flows.

All these methods above require sophisticated interface tracking or reconstruction techniques in order to obtain the desired results. Since our method is formulated with TP-LBM, it is not only much easier to solve, but also free from complicated tracking/reconstruction procedures to retain the intricate interface details, making our method particularly easy for complex scenarios.

Liquid simulation with multiphase phenomena. Although free surface methods are able to produce fascinating liquid interface phenomena, some multiphase effects, such as the bubble dynamics and the surface tension flows, require further modeling. Hong and Kim [23] modeled the discontinuous interface boundary condition for pressure and velocity gradients. Akinci et al. [9] proposed new force formulations to correctly address the surface tension and adhesion in SPH.

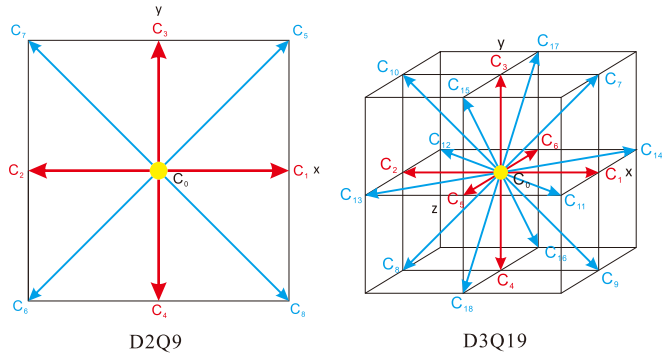


Fig. 1. Typical lattice structures in 2D (D2Q9) and 3D (D3Q19).

Losasso et al. [24] advanced particle level-set method to handle multiple liquid regions considering interaction. Kim [25] proposed to use regional level-sets for tracking a large number of fluid components. Boyd and Bridson [8] proposed to extend the FLIP framework to simulate the air and water dynamics. Cho and Ko [26] adopted the volume-of-fluid interface tracking method for simulating bubble dynamics. Solenthaler and Pajarola [27] modified the standard SPH method to simulate multiple fluids with moderate density ratio.

Note that these approaches adopt different models to simulate even a particular type of multiphase phenomena in liquid flows. However, our method is naturally unified and intrinsically involves all these effects in the same manner, including free surface flows, which enables our method for simultaneously simulating liquid-air flows with many multiphase phenomena involved. Moreover, our method is particularly unique to have stable simulations in turbulent circumstances, which is, to the best of our knowledge, the first work we are aware of that is based on TP-LBM.

3 Two-Phase Lattice Boltzmann Model

The lattice Boltzmann model (LBM) [28] is a mesoscopic computational model to approximate Navier-Stokes (NS) equations using the concept from continuous Boltzmann dynamics. LBM is highly parallel, and in many cases, can be an ideal alternative to the traditional NS-based approaches due to its simplicity and locality. LBM solves fluid flows by evolving the distribution functions f_i, which describe the probability of a particle with velocities $\mathbf{c}_i$ at position $\mathbf{x}$ and time t:

$$f_i(\mathbf{x}+\mathbf{c}_i\Delta t, t+\Delta t) = f_i(\mathbf{x},t) + \Omega_i(\rho,\mathbf{u}) + F_i(\mathbf{x},t)\Delta t, \quad (1)$$

where i indexes discretized variables; $\mathbf{c}_i$ is the lattice velocity; $\Omega_i(\rho,\mathbf{u})$ describes the change of distribution function due to particle collision based on macroscopic density ρ and velocity $\mathbf{u}$; $F_i(\mathbf{x},t)$ is the force term, and Δt is the time-step. Such equation is solved by a propagation and a collision step with discretized lattice velocity structures that align with the simulation grid. The typical lattice structures in 2D and 3D are D2Q9 and D3Q19 structures, respectively, as illustrated in Fig. 1. LBM is well-known for its simplicity due to linear differential operator and local boundary treatment, as well as the nice property for parallelism.

As a particular type of LBM, the two-phase lattice Boltzmann model (TP-LBM) can be used to describe the interface dynamics between gas and liquid, which is generally

classified into four categories in computational physics: pseudo-potential model [29], [30], color model [31], [32], mean-field model [13], [33], and phase-field model [14], [34]. In the literature of computer graphics, there are works [35], [36], [37] that are based on the phase-field model and a specific type of two-fluid model [38]. These works require large viscosity to stabilize the underlying dynamics, and are incapable of simulating complex turbulent interface flows at limited resolutions.

Our method, on the other hand, is the first work in graphics which is based on the *mean-field model* [13]. By experiments, we found that mean-field model can retain stable hydrodynamics with sharp interface, while other models have either high compressibility or thick interface that prohibit them from being employed for simulating complex turbulent interface dynamics. In the content that follows in this section, we introduce the fundamentals of the TP-LBM with the mean-field model.

TP-LBM with mean-field model. The TP-LBM we use is constructed with a continuous index function ϕ which classifies the two flow phases as well as their mixtures. The whole dynamics is governed by the following two sets of lattice Boltzmann equations by repeating Eq. (1), with the same lattice structure as shown in Fig. 1:

$$\begin{aligned} f_i(\mathbf{x}+\mathbf{c}_i\Delta t, t+\Delta t) &= f_i(\mathbf{x},t) + \Omega_i(\phi,\mathbf{u}) + F_{f,i}(\phi,\mathbf{u})\Delta t,\\ g_i(\mathbf{x}+\mathbf{c}_i\Delta t, t+\Delta t) &= g_i(\mathbf{x},t) + \Omega_i(\rho,\mathbf{u}) + F_{g,i}(\rho,\mathbf{u})\Delta t, \end{aligned} \tag{2}$$

where the first equation is called the index function equation and the second equation is called the momentum equation. The two equations are coupled by the macroscopic quantities ρ and $\mathbf{u}$; f_i is used to recover the index function ϕ; and g_i is used to calculate the pressure p and the momentum $\rho\mathbf{u}$:

$$\begin{aligned} \phi &= \sum_i f_i, \qquad p = \sum_i g_i + \frac{1}{2}\mathbf{u}\cdot\mathbf{F}_{ev}\Delta t,\\ \rho\mathbf{u} &= \frac{1}{c_s^2}\left(\sum_i \mathbf{c}_i g_i + \frac{c_s^2}{2}(\mathbf{F}_s+\mathbf{G})\Delta t\right), \end{aligned} \tag{3}$$

where $c_s = 1/\sqrt{3}$ is the speed of sound; $\Delta t = 1$; $\mathbf{G}$ is the external force such as the gravity; $\mathbf{F}_{ev}$ and $\mathbf{F}_s$ are the exclusion volume force for phase separation and the surface tension force respectively, which are calculated using the mean-field model as [12]:

$$\mathbf{F}_{ev} = -\nabla\psi(\rho), \qquad \mathbf{F}_s = \kappa\rho\nabla\left(\nabla^2\rho\right), \tag{4}$$

where κ controls the strength of surface tension; ψ is the difference between the pressure $\hat{p}$ from the non-ideal equation-of-state (EoS) and the pressure $p_0 = \rho c_s^2$ from the ideal EoS: $\psi = \hat{p} - p_0$. He et al. [12] proposed to use the Carnahan-Starling EoS (CS-EoS) particularly for LBM to calculate $\hat{p}$. Given specific thermodynamic parameters [13], the CS-EoS reduces to: $\hat{p}(\rho) = \rho c_s^2(1+\rho+\rho^2-\ \rho^3)/(1-\rho)^3 - 4\rho^2$, and the fluid density ρ is determined from the index function ϕ as follows:

$$\rho(\phi) = \rho_g + \alpha(\rho_l - \rho_g), \tag{5}$$

where ρ_g and ρ_l specify the density of the gas and liquid, respectively; α is the liquid concentration at each lattice point: $\alpha = (\phi - \phi_g)/(\phi_l - \phi_g)$, where ϕ_g and ϕ_l are lower and upper limits for the index function. It is worth mentioning that the TP-LBM only recovers hydrodynamics with small Mach number.

Collision model. The modeling of the particle collision Ω is important for both stability and accuracy, where the multiple-relaxation-time (MRT) model is employed with a relaxation matrix $\mathbf{\Lambda}$:

$$\Omega_i = -\Lambda_{ij}(f_j - f_j^{eq}). \tag{6}$$

Here, the summation is assumed for repeated indices, and f_i^{eq} is the equilibrium state. The MRT relaxation is defined by $\mathbf{\Lambda} = \mathbf{T}^{-1}\tilde{\mathbf{\Lambda}}\mathbf{T}$, where the orthogonal matrix $\mathbf{T}$ transforms the state vector $\mathbf{f}$ to the moment space, which is then relaxed by a new diagonal relaxation matrix $\tilde{\mathbf{\Lambda}}$, and subsequently transformed back by $\mathbf{T}^{-1}$. Thus, the relaxation is done in moment space with multiple relaxation time parameters [39], [40]. The specific forms of matrix $\mathbf{T}$ for D2Q9 and D3Q19 lattices are given in the appendix. The diagonal elements of $\tilde{\mathbf{\Lambda}}$, which form a relaxation vector $\mathbf{s}$, are linearly optimized parameters given in [39], [40], and the second element in $\mathbf{s}$ can be tuned to achieve a desired bulk viscosity. As the ordering of the moments is implicitly specified by $\mathbf{T}$, we can have the following ordered parameters in 2D:

$$\mathbf{s} = \{1.0, 1.14, 1.54, 1.0, 1.7, 1.0, 1.7, 1/\tau, 1/\tau\}, \tag{7}$$

and in 3D:

$$\begin{aligned} \mathbf{s} = \{&1.0, 1.14, 1.6, 1.0, 1.5, 1.0, 1.5, 1.0, 1.5, 1/\tau, 1.6,\\ &1/\tau, 1.6, 1/\tau, 1/\tau, 1/\tau, 1.18, 1.18, 1.18\}, \end{aligned} \tag{8}$$

where τ is the relaxation time determined from the kinematic viscosity:

$$\tau = \nu/(c_s^2\Delta t) + 1/2. \tag{9}$$

Note that a constant ν is usually used in the original TP-LBM model.

In TP-LBM with MRT collision model, the corresponding equilibrium distributions are defined as:

$$f_i^{eq} = \phi\Gamma_i(\mathbf{u}), \qquad g_i^{eq} = \rho c_s^2\Gamma_i(\mathbf{u}) + \Gamma_i(\mathbf{0})\psi(\rho), \tag{10}$$

with $\Gamma_i(\mathbf{u}) = w_i[1 + \mathbf{c}_i\cdot\mathbf{u}/c_s^2 + (\mathbf{c}_i\cdot\mathbf{u})^2/(2c_s^4) - \mathbf{u}\cdot\mathbf{u}/(2c_s^2)]$, where w_i are lattice coefficients, which are set according to different lattice structures [13]. The macroscopic quantities are defined in Eqs. (3) and (5), respectively.

Force model. The force model is used to maintain the sharp interface which is implicitly integrated, and the specific forms are:

$$F_{f,i} = f_i^F - \frac{1}{2}\Lambda_{ij}f_j^F, \qquad F_{g,i} = g_i^F - \frac{1}{2}\Lambda_{ij}g_j^F, \tag{11}$$

where

$$\begin{aligned} f_i^F &= \Gamma_i\frac{\mathbf{c}_i - \mathbf{u}}{c_s^2}\cdot\mathbf{F}_{ev},\\ g_i^F &= (\mathbf{c}_i - \mathbf{u})\cdot[\Gamma_i(\mathbf{F}_s+\mathbf{G}) + (\Gamma_i - w_i)\mathbf{F}_{ev}], \end{aligned} \tag{12}$$

with the definition of $\mathbf{F}_{ev}$ and $\mathbf{F}_s$ from Eq. (4). $\mathbf{F}_{ev}$ takes ϕ and ρ as the arguments for the index function and momentum updates. Note that in practical implementation with the

MRT model, the force should be in the moment space before relaxation, which is done by first assembling the force vector $\mathbf{f}^F$ and then converting it to the moment space vector $\mathbf{Tf}^F$.

Algorithm 1 gives a pseudo-code for the overview procedure of the TP-LBM with the mean-field model. *Readers can refer to [13] for more theoretical details.* These steps from the original TP-LBM serve as our base model, and could direct our GPU-based parallel implementation.

Algorithm 1. Pseudo-Code for TP-LBM Iteration

```
Initialize p, φ, ρ, u.
Initialize the distribution functions based on Eq. (10).
while 1 do
    Calculate φ and ρ (Eqs. (3) and (5)).
    Calculate ψ based on the definition.
    Calculate F_s and F_ev (Eq. (4)).
    Calculate u (Eq. (3)).
    Calculate p using the computed u (Eq. (3)).
    Compute collision (Eq. (6)) and propagation (Eq. (2)).
end while
```

Determination of density ratio. According to [13], the Maxwell construction from the thermodynamics gives the equilibrium densities of the two phases at a fixed temperature, which are numerically solved to be $\phi_g = 0.02255$ and $\phi_l = 0.25008$ respectively, with the parameters of the CS-EoS chosen to balance the overall stability and the interface thickness. Ideally, ρ_g and ρ_l can be arbitrarily chosen irrespective of ϕ_g and ϕ_l. But practically, they should be equal ($\rho_g = \phi_g$ and $\rho_l = \phi_l$) for the reason of numerical stability.

To change the density ratio, it would require finding another set of parameters for the CS-EoS, which is very difficult to achieve with the current model. In this paper, we fix the density ratio to be ϕ_l/ϕ_g. One may argue that this density ratio is not true for practical cases. However, since we target for graphical flow simulation, we found that such density ratio already gives visually acceptable animation results and could be adopted in our simulation.

4 Our Method

The above TP-LBM with the mean-field model is a single-component simulation model that can already be used for liquid-air interface flow simulations, since the dynamic behavior of air can be approximated by the liquid vapor, which simplifies the dynamics with the acceptable visual realism. However, the TP-LBM with the mean-field model can hardly survive in turbulent circumstances due to strong instability. By extensive experimental analysis, we found that the oscillation is the main source of instability for the original model, which generally comes from the intrinsic weak compressibility related to the local pressure update of the model. The reason is that all the fields are coupled together in TP-LBM, and the velocity magnitude can easily go out of range by the erroneous compression wave, which eventually breaks down the simulation.

To increase the stability, we propose a set of techniques that are combined together to suppress the oscillation. Moreover, to enable the simulation at high Reynolds number, we also derive a new subgrid-scale model for the oscillation-suppressed TP-LBM, which is based on the observation that the sharp interface change is responsible for the instability with the presence of large shear flows. These modifications to the TP-LBM with the mean-field model make the entire simulation stable enough for turbulent liquid-air interface flow dynamics in a variety of scenarios, simultaneously involving multiple multiphase phenomena within one computational model.

4.1 Oscillation Suppression

Since the TP-LBM permits weak compression, the pressure field does not necessarily satisfy the incompressibility condition, and is susceptible to large errors by the oscillatory compression waves. These oscillations may generate significant overshoots of physical quantities that accumulate to finally form a very large velocity to break down the simulation, especially for turbulent interfaces.

It is important to note that the overshooting problem in TP-LBM differs from those generated by the numerical instability of the traditional PDE solver, as the TP-LBM simulates hydrodynamics near equilibrium, which naturally consists of density perturbations susceptible to pressure waves. When such perturbation is strong enough, it is very likely to break down the simulation, since the velocity defined in Eq. (3) is sensitive to density perturbations. Thus, we propose the following techniques to suppress the oscillation in the framework of the TP-LBM with the mean-field model, so that the complex liquid-air interface flows can be well simulated.

Viscosity interpolation. Recall that in the original model (Section 3), a constant kinematic viscosity is used, which must be set small for turbulence, where large flow velocity arises to cause instability. In addition, using constant kinematic viscosity is not physically consistent, since air has much different kinematic viscosity than liquid (such as water). Thus, a spatially varying kinematic viscosity should be determined, especially for turbulent interface flows where the viscous stress influences greatly the overall stability. Experimentally, we found that the reciprocal interpolation [41] for the local viscosity can perform well for high Reynolds number flows:

$$\nu = 1/\left[(1-\alpha)\nu_g^{-1} + \alpha\nu_l^{-1}\right], \tag{13}$$

where at each lattice point, the kinematic viscosity is interpolated based on the liquid concentration α, instead of using a constant value across the whole domain. We adopt this viscosity scheme for all our simulations, which has the ability of avoiding too large velocity to create compression waves due to improper viscosity values. Here, the value of ν in Eq. (13) will be used in Eq. (9) to evaluate the relaxation time τ.

As a method of viscosity modeling for different phases, the turbulence can be suppressed in the air region, increasing the overall stability, while not excessively smoothing the interfaces. However, the non-interface region has almost constant viscosity, which is still not very stable in turbulent cases, although the vortices in these regions don't usually exhibit complex features.

Clipping scheme. In the TP-LBM, the mass is traced by the index function ϕ, which is employed to determine the density and viscosity by the interpolation using the liquid concentration α (see, Eqs. (5) and (13)). In the equilibrium case,

the interpolated fields are within the prescribed range: $\rho \in [\rho_g, \rho_l]$ and $\nu \in [\nu_l, \nu_g]$ ($\rho_g < \rho_l$, $\nu_l < \nu_g$). However, in the dynamic case, α may go beyond the range $[0, 1]$, leading to very small or even negative density and viscosity, which may be unphysical and cause overshoots to break down the simulation. Therefore, we propose to clip α within the range $[0, 1]$ prior to interpolating ρ and ν. Note that this only affects the momentum equation, and ϕ is allowed to fully recover the original continuity equation. Thus, the original property of the mass conservation in the TP-LBM is not affected.

Pressure treatment. In the original work of [13], the index function and momentum equations share the same definition of ψ (see, Eqs. (2), (11) and (12)), where the same pressure force $\hat{p}$ from the CS-EoS is applied, which is quite unstable in turbulent cases. In fact, the index function equation is stable by using the original definition of ψ since it only takes effect along the interface where thermodynamic pressure should be used. However, for momentum equation, using thermodynamic pressure again is no longer appropriate since a small thermodynamic perturbation in the pressure will significantly influence the stability of the hydrodynamics, especially along the interface.

Thus, we propose a new pressure treatment method similar to that by Lee and Lin [33] so that the TP-LBM stability can be significantly increased for liquid-air simulation. Since the momentum equation recovers the hydrodynamics, we regard the pressure p in Eq. (3) as the hydrodynamic pressure, and use it consistently for all the terms involving pressure in the momentum equation.

Specifically, the pressure used in equilibrium state g_i^{eq} should first be replaced by the hydrodynamic pressure p, which redefines ψ for the momentum equation as $\psi(\rho) = p - \rho c_s^2$. Second, the force definition in the momentum equation should become $\mathbf{F}_{ev}(\rho) = -\nabla(p - \rho c_s^2)$, where p is also the hydrodynamic pressure. Although the hydrodynamic pressure is generally smooth over the whole field compared to the thermodynamic pressure, it can still be influenced by the compression errors. As Lee and Lin analyzed [33], without such compression error, the hydrodynamic pressure gradient has much lower order than the density gradient and can be neglected. Thus, we can drop the pressure p in the force $\mathbf{F}_{ev}(\rho)$ to eliminate the compression errors and thus can increase stability apparently. Therefore, Eq. (12) is finally formulated as $\mathbf{F}_{ev}(\rho) = c_s^2 \nabla \rho$.

It is worth mentioning that the temperature is not a key component in our framework since the TP-LBM simulates hydrodynamics assuming isothermal conditions. However, our model is two-phase, and the interface transition inevitably requires certain thermodynamic formulation, such as the EoS. This is where the temperature comes from.

4.2 Detail Preservation

Using the above techniques to suppress the oscillation, the stability has been significantly improved over the original model, but the viscosity must be set sufficiently large, which can only produce relatively smooth interface unless we use a very high resolution grid. As mentioned before, LBM requires low Mach number for hydrodynamic flows. Thus, to generate flows with turbulence details, a larger Reynolds number must be achieved with lower viscosity under the condition of limited resolution and restricted velocity range. To allow small enough viscosity value, the Smagorinsky model is traditionally used for LBM simulations [42] to resolve subgrid-scale interactions. Such treatment is a common practice in LBM to stabilize the flow in turbulent situations by introducing an eddy viscosity while recovering a reasonable mean flow. However, the model coefficient for the eddy viscosity is usually empirically set, and can be locally over-predictive or under-predictive [43]. We derive a new subgrid-scale model that depends on a momentum-based filtering technique motivated from the dynamic Smagorinsky model [44] for automatic determination of local model coefficients, which considers the density variation and provides more appropriate diffusion at the interface, so that much higher Reynolds number flow simulations can be achieved.

Model definition. We derive our subgrid-scale model from the algebraic model in the large eddy simulation (LES) due to simplicity. Standard algebraic LES models such as the Smagorinsky model and its variants are not appropriate for our TP-LBM since they do not consider the variation of density and are less stable in our case. To start, we first consider the macroscopic momentum equation that the TP-LBM recovers [13]:

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla p + \nu \nabla^2 (\rho \mathbf{u}) + \mathbf{F}, \tag{14}$$

where $\mathbf{F}$ is the force term containing both intermolecular and external forces. If we use $\overline{(\cdot)}$ to denote the spatial filtering at the grid level and define the momentum as: $\mathbf{m} = \rho \mathbf{u}$, with a given grid resolution, the macroscopic momentum equation is filtered as:

$$\frac{\partial \overline{\mathbf{m}}}{\partial t} + \nabla \cdot \frac{\overline{\mathbf{m}\mathbf{m}}}{\overline{\rho}} = -\nabla \overline{p} + \nu \nabla^2 \overline{\mathbf{m}} + \overline{\mathbf{F}} - \nabla \cdot \sigma, \tag{15}$$

where we ignore the subgrid-scale effects of the force term and

$$\sigma = \overline{\mathbf{m}\mathbf{m}/\rho} - \overline{\mathbf{m}}\ \overline{\mathbf{m}}/\overline{\rho} \approx (\overline{\mathbf{m}\mathbf{m}} - \overline{\mathbf{m}}\ \overline{\mathbf{m}})/\overline{\rho}, \tag{16}$$

assuming ρ can be taken out of the filter, which is almost exact for flows far from the interface, and approximate for flows around the interface. Note that we use the density and momentum as the filtered quantities since the TP-LBM only resolves them. For the non-interface regions, filtering the momentum works almost essentially the same as filtering the velocity, while around the interface, this introduces appropriate amount of diffusion to gain more stability. Experimentally, we found that the approach of modeling sub-grid scale effects by the idea of filtering works reasonably well in the context of graphics, which often favors stability over physical accuracy. Following the similar form as in the standard Smagorinsky model, we then define our symmetric momentum-based Reynolds stress tensor as:

$$\tau = \overline{\mathbf{m}\mathbf{m}} - \overline{\mathbf{m}}\ \overline{\mathbf{m}}, \tag{17}$$

and model it to be:

$$\tau_{ij} - \frac{1}{3}\tau_{kk}\delta_{ij} \approx -2C\Delta^2 |\overline{\mathbf{S}}| \overline{S}_{ij}, \tag{18}$$

where the deformation tensor is defined based on the momentum as:

$$\mathbf{S}(\mathbf{m}) = \left[\nabla\mathbf{m} + (\nabla\mathbf{m})^{\top}\right]/2. \tag{19}$$

The deformation tensor $\mathbf{S}$ is symmetric by definition and traceless for incompressible flow. The gradients are computed by finite-difference at the lattice grid; $|\cdot|$ is the tensor norm, which is defined as:

$$|\mathbf{S}| = \sqrt{2S_{ij}S_{ij}}; \tag{20}$$

and C is the model coefficient. It is clear that the model in Eq. (18) matches in symmetry and dimension, which is physically consistent.

Traditionally, the Boussinesq approximation tells us that the Reynolds stress is modeled to be linear to the deformation tensor: $\tau_{ij} - \frac{1}{3}\tau_{kk}\delta_{ij} \approx -2\nu_t\overline{S}_{ij}$, where ν_t is termed the eddy viscosity. Combining Eqs. (16), (17) and (18), we find the eddy viscosity for our TP-LBM model as:

$$\nu_t = \frac{C\Delta^2|\overline{\mathbf{S}}|}{\overline{\rho}}. \tag{21}$$

Note that the above filter is not explicitly applied. Rather, it is implicitly done by a given grid resolution. More specifically, the variables marked with $\overline{(\cdot)}$ are taken from the simulation grid directly. Thus, for the purpose of clarity, we remove the implicit filtering symbol $\overline{(\cdot)}$ in the subsequent derivations.

Coefficient evaluation. The model coefficient C can be empirically set, but this is inappropriate since many existing experiments verify that C should vary with location and time. Following the similar procedure as in the dynamic Smagorinsky model, to evaluate C, we assume scale invariance of the turbulence model, and the coefficient C can be determined by the similar identity across different scales. To realize, we first apply a test filter $\widetilde{(\cdot)}$ to Eq. (15) with twice the size of the grid spacing: $\tilde{\Delta} = 2\Delta$, which is constructed by extending a 1D box filter with the filter weight coefficients {0.25, 0.5, 0.25} to 3D [45]. Then, a new Reynolds stress tensor appears after such filtering:

$$\tau' = \widetilde{\mathbf{m}\mathbf{m}} - \widetilde{\mathbf{m}}\widetilde{\mathbf{m}}, \tag{22}$$

which can be modeled by the assumed scale invariance and repeating Eq. (18) to obtain:

$$\tau'_{ij} - \frac{1}{3}\tau'_{kk}\delta_{ij} \approx -2C\tilde{\Delta}^2\left|\widetilde{\mathbf{S}}\right|\widetilde{S}_{ij}. \tag{23}$$

Meanwhile, the Germano's identity [44] can be established:

$$\tau'_{ij} - \widetilde{\tau_{ij}} = \widetilde{m_i m_j} - \widetilde{m}_i\widetilde{m}_j, \tag{24}$$

which is derived from Eqs. (17) and (22). By further using the Eqs. (18) and (23), we arrive at another approximation:

$$\mathbf{L} \approx -2C\Delta^2\mathbf{M}, \tag{25}$$

where

$$\begin{aligned} L_{ij} &= \widetilde{m_i m_j} - \widetilde{m}_i\widetilde{m}_j, \\ M_{ij} &= 4\left|\widetilde{\mathbf{S}}\right|\widetilde{S}_{ij} - |\widetilde{\mathbf{S}|S}_{ij}. \end{aligned} \tag{26}$$

Note that both the left and right hand sides of Eq. (25) are computable from the resolved quantities, and all the tensor elements share the same parameter C. The most straightforward approach for determining C is to minimize the approximation error in Eq. (25) with respect to C using the least-square fitting [46]. Assuming summation over the repeated indices, the error is defined as:

$$E(C) = (L_{ij} + 2C\Delta^2 M_{ij})^2, \tag{27}$$

and the model coefficient is solved from the following minimization problem:

$$C = \underset{C}{\operatorname{argmin}}\ E(C) = -\frac{1}{2\Delta^2}\frac{L_{ij}M_{ij}}{M_{ij}M_{ij}}. \tag{28}$$

Apply to TP-LBM. Recall from Eqs. (7) and (8) that the MRT collision model used in our TP-LBM requires a relaxation time τ to determine the relaxation parameters, which is computed from the kinematic viscosity ν. With our eddy-viscosity-based subgrid-scale model, there is an effective viscosity ν_{eff} which is defined as: $\nu_{\text{eff}} = \nu_0 + \nu_t$. Then, the relaxation times in Eqs. (7) and (8) are replaced with the effective viscosity in order to take into account the turbulence details, which will be reflected on the interface. Algorithm 2 gives a pseudo-code for the parallel computation of the relaxation time using our subgrid-scale model.

Algorithm 2. Pseudocode of Our Subgrid-Scale Model

1: **for all** fluid lattice points in parallel **do**
2: Calculate $\mathbf{S}$ from the lattice variables using the 2nd order central difference, and store the related elements at each lattice point (Eq. (19)).
3: Compute the tensor norm $|\mathbf{S}|$ and store it at each lattice point (Eq. (20)).
4: **end for**
5: **for all** fluid lattice points in parallel **do**
6: Locally filter $\mathbf{S}$ to obtain $\widetilde{\mathbf{S}}$.
7: Locally filter $|\mathbf{S}|\mathbf{S}$ to obtain $\widetilde{|\mathbf{S}|\mathbf{S}}$.
8: Compute filtered tensor's norm $\left|\widetilde{\mathbf{S}}\right|$.
9: Compute $\mathbf{M}$ using $\left|\widetilde{\mathbf{S}}\right|$, $\widetilde{\mathbf{S}}$, and $\widetilde{|\mathbf{S}|\mathbf{S}}$ (Eq. (26)).
10: Locally filter $\mathbf{m}$ to obtain $\widetilde{\mathbf{m}}$, which constructs $\widetilde{\mathbf{m}}\widetilde{\mathbf{m}}$.
11: Locally filter $\mathbf{m}\mathbf{m}$ to obtain $\widetilde{\mathbf{m}\mathbf{m}}$.
12: Compute $\mathbf{L}$ using $\widetilde{\mathbf{m}\mathbf{m}}$ and $\widetilde{\mathbf{m}}\widetilde{\mathbf{m}}$ (Eq. (26)).
13: Compute C using $\mathbf{L}$ and $\mathbf{M}$ (Eq. (28)).
14: **end for**
15: **for all** fluid lattice points in parallel **do**
16: Locally filter C to remove noise.
17: Compute eddy viscosity ν_t using $|\mathbf{S}|$ (Eq. (21)).
18: Compute the relaxation time τ using the effective viscosity ν_{eff}.
19: **end for**

5 Results

By using the techniques presented above, we can simulate complex liquid-air interface flows. The proposed framework was implemented on a computer with an Intel Xeon E5-2630 v3 CPU and 32 GB system memory. Our method is highly parallel and is implemented on an NVIDIA GTX TITAN X GPU, and the simulations may take at most

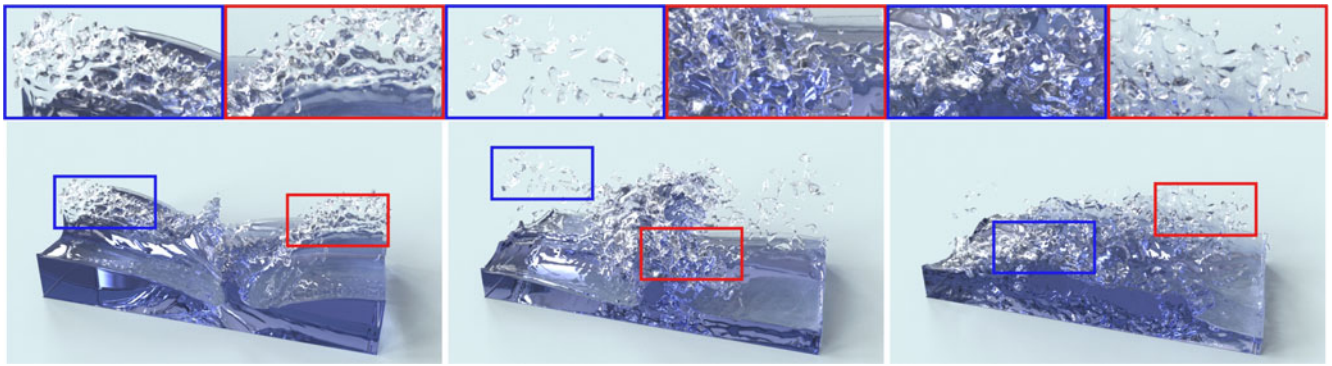


Fig. 2. Simulation of a quadruple dambreak demonstrating splashing due to interface turbulence (grid resolution: $400 \times 200 \times 200$).

around 6 GB on-board memory at the highest resolutions such as $256 \times 256 \times 256$ or $400 \times 200 \times 200$. According to different purposes of animations, the TP-LBM iteration per frame may also vary. However, the performance of the TP-LBM algorithm is constant and presents good scalability on parallel machines. To render the flow, we employ the marching cubes algorithm [47] to extract the isosurface from the density field as the interface by tuning appropriate isovalues. We also implemented a GPU-based path tracer [48] to render all the shown results.

Parameter setting. We use a normalized configuration in our simulation as in many lattice Boltzmann simulations. κ is set according to different conditions; To extract the interface isosurface, we tune the isovalue for the density concentration within the range $[0.2, 0.8]$, and it is usually set as 0.8. Note that the variation of ρ takes place only in a small band around the interface (usually across three lattice points). The gravity $\mathbf{G} = -\rho g \mathbf{n}_y$ is calculated in lattice units based on the characteristic velocity v_c and the domain height L_h [12]: $g = v^2/L_h$, so that the same characteristic velocity can be retained for different grid resolutions.

Initialization and boundary treatment. It should be noted that the TP-LBM is only valid when the distribution functions are not far from equilibrium. To ensure this, the initial velocity field must first satisfy the low Mach number condition, which is then used for initializing the distribution functions by calculating the equilibrium state. In addition, the initial pressure profile needs to satisfy hydrodynamic condition, which ideally requires solving a Poisson equation as in the traditional Navier-Stokes solver. However, for simplicity, we analytically select a pressure gradient profile to be as close as to the ideal solution in each case, and our TP-LBM can automatically evolve the pressure to the expected profile within very few iterations.

To enable no-slip (zero velocity) boundary condition at the wall, such as the fluid container, we employ half-way bounce-back scheme [49], which also ensures mass conservation. In this scheme, particles reaching the boundary are reflected (bounced back) between the fluid and the wall lattice points, which is incorporated in the propagation step.

Free surface flow. We use the dambreak to demonstrate the free surface liquid flow, where the liquid volumes are initially static, and the initial hydrodynamic pressure is set to zero for the whole domain. The pressure propagation is so fast that appropriate hydrodynamic pressure field can be quickly formed without introducing significant errors for stable simulations thereafter. We use a $400 \times 200 \times 200$ simulation grid, and a high Reynolds number is reached by setting small viscosities: $\nu_g = 0.01$ and $\nu_l = 0.00001$. The surface tension κ is set to 0, which is reasonable in a large-scale scene where kinetic energy dominates the surface energy, and the surface tension can be ignored. At characteristic velocity $v_c = 0.1$, we use 120 iterations for each animation frame, and each frame takes 45 seconds for the simulation on our platform. Fig. 2 shows the snapshots of our simulation result for the dambreak scene. Note the fine splash details.

In addition, we also show a pool of liquid using a resolution of $300 \times 150 \times 300$. The liquid is initialized with an arbitrarily designed geometry. In this simulation, we use the same characteristic velocity $v_c = 0.1$ but 100 iterations per video frame, and set $\nu_g = 0.1, \nu_l = 0.00005$ and $\kappa = 0.00005$. The snapshots are shown in Fig. 3.

With our TP-LBM scheme, we can observe splitting and merging processes of liquid collisions, which create sufficient interface details with well-captured splashes. Our TP-LBM respects local mass-conservation in streaming, collision and boundary conditions, thus allowing global volume conservation, no matter how long the simulation goes. The interface thickness does not vary with grid resolutions, which is an intrinsic property for our method that suggests using finer grids to capture relatively thinner and better interface, which

Fig. 3. Simulation of a pool of liquid with arbitrary initial geometry (grid resolution: $300 \times 150 \times 300$).

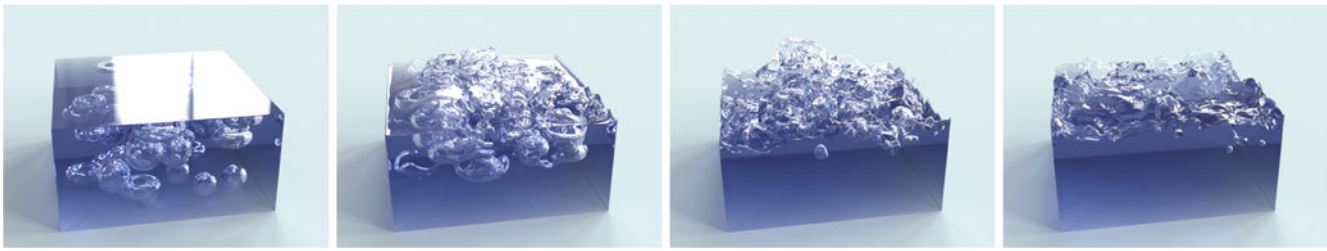


Fig. 4. Simulation of bubble flows inside a liquid pool with complex interactions (grid resolution: $256 \times 256 \times 256$).

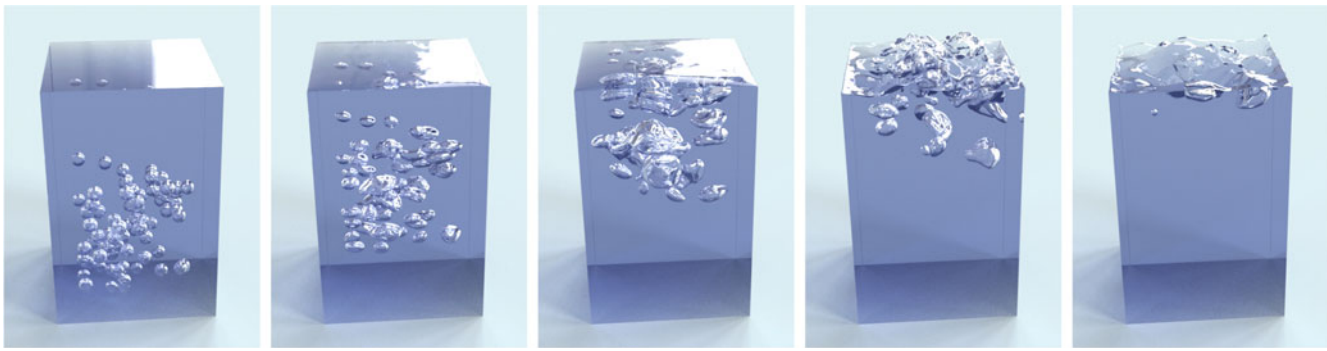


Fig. 5. Simulation of many small bubbles rising in a liquid (grid resolution: $200 \times 400 \times 200$).

Fig. 6. Simulation of surface-tension-dominated liquid flow with zero gravity (grid resolution: $200 \times 200 \times 200$).

is still computationally efficient due to high parallelism of our method. However, small features may diffuse into the ambient because of the diffuse interface representation, which is a drawback of the proposed method.

Bubble flow. In bubble dynamics, we initialize the static bulk volume with a set of randomly distributed bubbles in different sizes. It should be noted that the weak compressibility of the TP-LBM may lead to the liquid volume shaking if the initial values are not properly given. To have stable dynamics, we assume an initial state with a hydrostatic balance at the given gravity. The initial pressure is set by employing the hydrostatic relation $\partial_h p = \rho g$, and the pressure gradient for the gas phase is ignored. This allows the open surface to remain sufficiently calm before bubbles emerge. It is also important to include sufficient surface tension force, since in typical cases, bubbles have small sizes and the surface tension keeps a smooth shape of bubbles. In our $256 \times 256 \times 256$ simulation, we set $\kappa = 0.0001, \nu_g = 0.005, \nu_l = 0.00005$. The characteristic velocity v_c is set to 0.1 and each frame consists of 120 iterations. Fig. 4 demonstrates the rising and inter-collision of bubbles, where the intricate dynamics are generated.

We also carried out a simulation of a tall tank of still liquids initially filled with many small bubbles. The bubbles move upwards under the buoyancy force, while merging and twisting to form larger bubbles. For even better incompressibility, the gravity is halved for our tall tank simulation and each frame is made from 240 iterations at the resolution of $200 \times 400 \times 200$; other parameters are set as: $\nu_g = 0.02, \nu_l = 0.00005, \kappa = 0.0002$. See, Fig. 5 for a demonstration. Note that the liquid volume remains still, as a result of both mass conservation and incompressibility.

Surface tension flow. In surface-tension-dominated interface flows, we demonstrate a case with zero gravity, and the liquid volumes are initialized with sharp features. Due to the surface tension force, the interfacial surface develops to gradually reach a smooth shape while respecting volume conservation. This can be easily done by our method without any modification since the surface tension force is calculated in a unified manner, and thus can be tweaked in a considerable range. Fig. 6 shows a process where the smooth liquid volumes are recovered from an initial shape with sharp corners. Note that different κ values may result in different recovery processes. The grid resolution is $200 \times 200 \times 200$ and each frame has 80 iterations.

Falling liquid with all effects. Finally, to demonstrate the powerfulness as well as the difference of our method as compared to the existing techniques, we demonstrate an example of falling liquids which includes simultaneously all the above effects in one example, see, Fig. 7. In this example, we intentionally increase the surface tension so that the liquid becomes round on the separating plates in order to significantly show

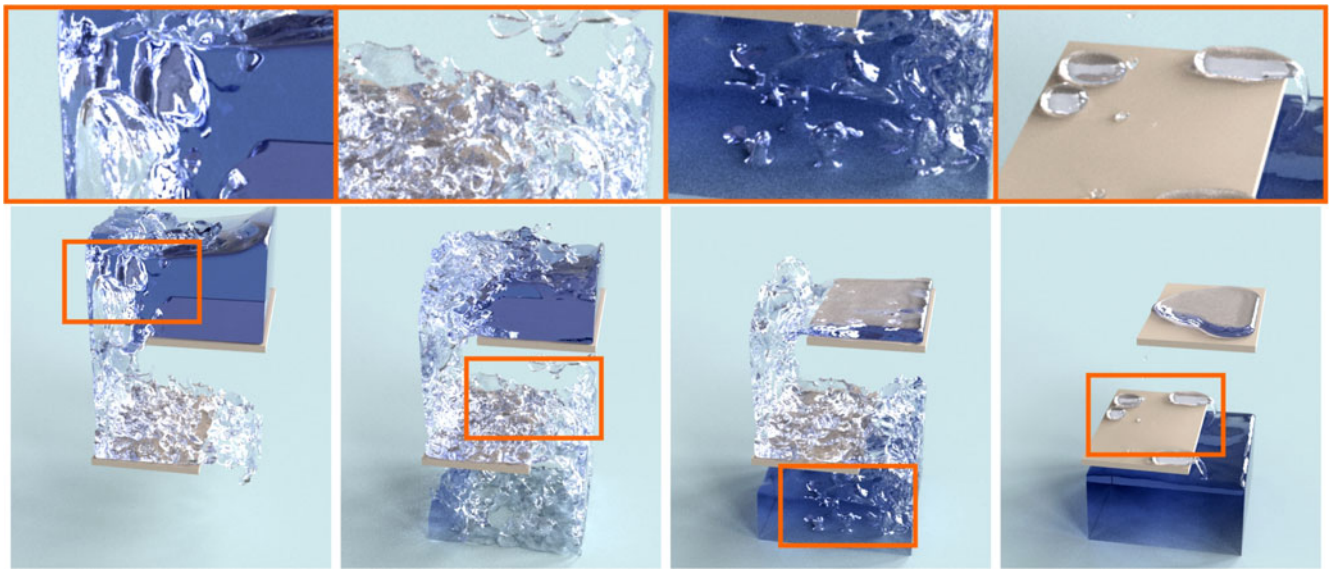


Fig. 7. Simulation of a tank of liquid pouring down and passing through two separating plates. The plates are set to be hydrophobic and have curved edges to initiate instability. This example demonstrates simultaneously the free surface turbulence, bubble formation, and surface tension effects, which are simulated within the same computational framework (grid resolution: $200 \times 400 \times 200$).

the difference of our method. The plates are made hydrophobic by setting the wall density to that of the gas phase in order to control the force at the boundary. The grid resolution in the simulation is set to $200 \times 400 \times 200$, and the plates are placed in height at $0.3N_y$ and $0.7N_y$ respectively, with $0.65N_x$ width and with curved edges. Each frame consists of 160 iterations with the characteristic velocity $v_c = 0.1$, taking up around 60 seconds on our platform. It is clear that bubbles appear automatically in a free-surface flow.

Note that this example is difficult to be achieved with the most usual free-surface methods, such as the Fluid-Implicit Particles method [5]. The result of comparing with the FLIP method is shown in Fig. 8. In the comparison, we adopted the MantaFlow framework [50] to carry out the FLIP simulation. The provided adaptive-time-stepping open-surface scene configuration of MantaFlow is modified with a larger CFL number of 5, and fixed by disabling the particle re-sampling procedure, which would lead to extreme volume gain in this scene setup. Other minor differences include that the plates for FLIP are with flat edges, and that the initial volume is separated from the ceiling by 2 percent of the scene height, in order to avoid the total sticking. Such a configuration in MantaFlow provides an efficient solution for FLIP-based simulation while not introducing too much error. It is obvious, however, that the air interaction is absent, and the volume loss over a long time is noticeable by the observation of the last row of the comparison in Fig. 8, even though the particle number is kept constant during the simulation.

We set the grid resolution of FLIP as $150 \times 300 \times 150$ and each frame takes about 53 seconds to simulate on CPU by the provided parameters, which is very close to our simulation of 60 seconds per frame on GPU, as we would like to carry out a comparison with similar computational time per frame. By working out the physical dimensions using $L_h = 1\,\mathrm{m}$ and $g = 9.8\,\mathrm{m/s^2}$, the respective time-steps are $0.0105\,\mathrm{s}$ for FLIP and $0.0128\,\mathrm{s}$ for our method. This means that the two simulations are of approximately the same time scale as shown in our video.

It is noticeable in the first two rows of Fig. 8 that our TP-LBM method does not seem to keep up with the pace of the FLIP method. However, this is naturally caused by the interaction between the air and the liquid, which would mix together to form rising bubbles to slow down the flux rate of the falling liquid, as compared to the result of the FLIP method. This phenomenon is also referred to as the glugging effect. The improvement of the FLIP method by Ando et al. [51] considers air incompressibility and a similar comparison is made to its free surface counterpart. Because air exists in our model, bubbles are naturally formed, and would appear even at the bottom of the container, which are missing for the FLIP method (see, the third row of Fig. 8). Additionally, we use a large surface tension and non-wetting plates to demonstrate the behavior of the liquid at smaller physical scale where the surface tension plays an important role in many interesting phenomena, such as the Plateau-Rayleigh instability.

Note that our solver is naturally vectorizable on GPU for easy parallelism, which is more reasonable to have a GPU-based implementation. However, MantaFlow only provides CPU-based FLIP simulations (with multi-threading enabled), which is hard to be shifted to GPU. For fair comparison, we have taken the advantage of the GPU, since it is not fair to implement a parallel method serially and compare with a non-parallel method but serially optimized. By tuning parameters to reach similar computational time, our method produces comparable interface details as the FLIP method. However, since our method can capture multiphase phenomena simultaneously, bubbles and surface tension effects can co-exist while they completely disappear in the FLIP method.

6 Discussion

Interface capturing. Our TP-LBM model is a diffuse interface model which contains liquid-air mixture around the interface. Essential to this scheme is the diffuse but non-dissipative

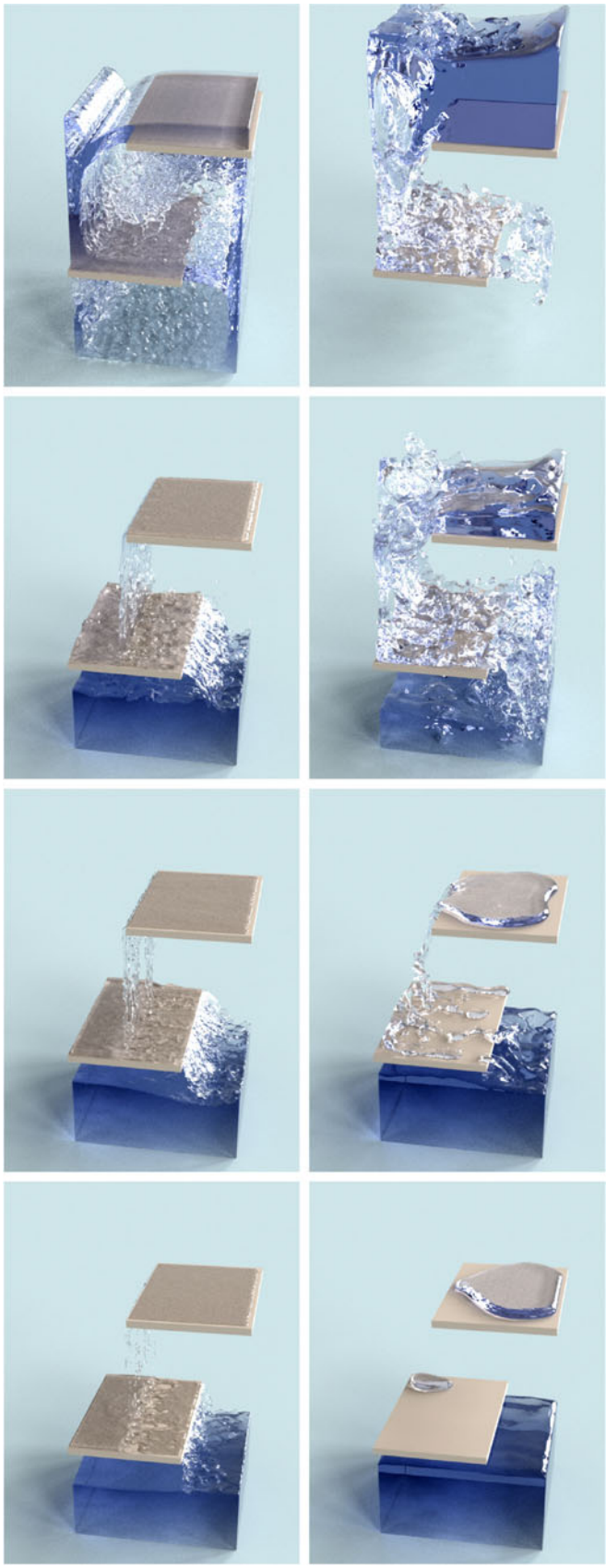


Fig. 8. Comparison of the FLIP method (left) with our method (right) for the liquid simulation. From the top to the bottom rows, frames 62, 228, 400 and 599 are shown, and their time-steps are closely matched. The significant visual difference is caused by the presence of the air phase and the surface tension at the interface.

interface maintained by the interface forces, without the need of explicit treatment for reconstruction, like for level-sets [3]. The interface transitions across a very thin band (typically two or three lattice points) regardless of the grid resolution. The interface geometry can be approximated by identifying the isosurface of ϕ, which can be easily achieved by employing the existing marching cubes algorithm [47].

The advantage of our interface capturing approach over the traditional level-set-based interface tracking methods is that such capturing scheme is incorporated into the dynamics, and the mass density is easily conserved as a nice property of the TP-LBM collision operator. In contrast, mass conservation has always been a critical issue for traditional methods, where fluid mass is not mathematically conserved, leading to obvious volume change. In addition, our interface capturing is extremely easy for multiphase dynamics using only simple updating rules of LBM.

However, our model does not prevent phase transition. Thus, small droplets that span less than ten lattice points or so, including the interface, may evaporate into the ambient gas phase, as the interface thickness cannot be ignored in the force evaluation, which requires a finite difference stencil to compute. The total mass of the fluid is balanced by the condensation of the gas phase into the liquid phase in other regions of the entire fluid domain.

Our model constructs the exclusive volume force $\mathbf{F}_{ev}$ and the surface tension force $\mathbf{F}_s$ independently. $\mathbf{F}_{ev}$ is responsible for phase separation and maintaining the interface, which is enough to hold the droplets together without the surface tension force. Therefore, the surface tension force can be tweaked down to zero to allow highly turbulent flows. However, it is desirable to keep a small amount of surface tension for higher stability when the Reynolds number is very large. On the contrary, in the TP-LBM with phase-field model [14], [34], in order to have stable simulation, the surface tension force could not be too small, thereby reducing the interface details.

Stability. Since our TP-LBM recovers an incompressible Navier-Stokes equations only asymptotically in the low Mach number limit, when the flow velocity is large, the compressibility may generate numerical waves to deteriorate the entire system stability, or damage the visual results. With our technique, numerical waves are sufficiently suppressed and the stability is much improved even for higher Mach number flows, which enables faster simulation. Our subgrid-scale model also gives adaptive diffusion to where it is likely to be unstable, which further enhances the stability for simulating very high Reynolds number flows.

Subgrid-scale model. Note that our subgrid-scale model is not a physically accurate model for interface flow dynamics based on the TP-LBM. Rather, it is developed with some empiricism from the fundamental concept and assumptions of the dynamic Smagorinsky model by considering the variation of density, which is more reasonable. The model is algebraic and easy to compute, but very effective in producing believable interface details, and also further stabilizes the entire system to a very small viscosity value.

Comparison. With our method, the stability can be significantly improved over the TP-LBM with the mean-field

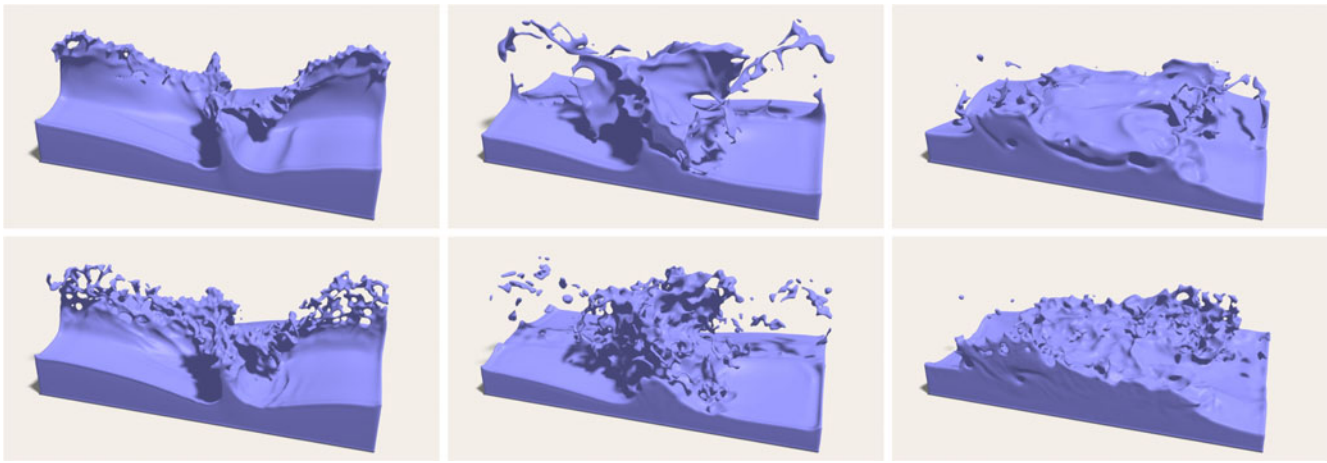


Fig. 9. Comparison with [13] at high Reynolds number. Top: the original TP-LBM with the mean-field model. Bottom: our method. The original TP-LBM with the mean-field model cannot support high enough Reynolds number to create fine-enough interface details due to high instability of the model (grid resolution: $300 \times 150 \times 300$).

model, and the subgrid-scale stress can be more properly added by calculating a spatially and temporally varying effective viscosity, which enhances details at the interface. Meanwhile, our method gives relatively larger effective viscosity close to the interface where the density and velocity vary significantly, which further improves the system stability. For clarity, Fig. 9 compares the results between the simulations from [13] and ours. The top row shows the snapshots of the simulation results from [13], where the Reynolds number is set as high as possible (3×10^3). The bottom row shows the simulation results with our method. Since our method is more stable, a much higher Reynolds number (7.5×10^5) can be reached, with intricate interface details. The Reynolds number in this comparison is defined as $Re = v_0 N_y / \nu_l$, and ν_g is fixed to 0.005, while $v_0 = 0.1$ and $N_y = 150$.

We want to emphasize that our model may not reach the performance and quality of the state-of-the-art solvers for each individual phenomena, such as the free surface flow simulators using particle-based methods [20]. However, our model is unique since it is unified to incorporate all different kinds of multiphase phenomena simultaneously under the turbulent conditions, which is desirable for many practical liquid animations.

On the other hand, the existing multiphase solvers often track a sharp interface between multiple fluid components, and apply a specific modeling of the forces at the interface in order to have multiphase phenomena, such as the surface tension effects, which are more complex [8], [23], [24], [26]. In comparison, our method applies universal forces to all fluid regions without tracking, allowing easy implementation. More importantly, the forces automatically take effect at the interface to capture multiple phenomena within the same computational framework.

Limitations. Since the density ratio in our model is determined by the EoS rather than the pure mechanical models, it is not straightforward to tune the density ratio directly as a prescribed value. The artifacts due to the relatively low density ratio may be noticeable, such as the slower falling droplets, as well as some phase transition phenomena of small droplets. The compressibility of the TP-LBM also prevents the simulation with very large gravity.

7 Conclusion

We propose in this paper a unified framework for liquid-air flow dynamics simulation. We explore the plausibility of the lattice Boltzmann method for simulating complex liquid-air interface phenomena in the graphics context. A unique feature of the framework is an interface capturing scheme which does not require complicated tracking or reconstruction procedures. By developing from the TP-LBM with the mean-field model using techniques in different aspects, together with a new density-aware subgrid-scale model, the whole simulation framework can capture intricate fine interface details to preserve realism while having sufficient system stability. Future work on more stable and accurate collision models with better interface force formulations may further enhance the system stability, with higher density ratio achieved to handle more challenging cases.

Appendix

Following the ordering of the lattice velocities shown in Fig. 1, we provide the **T** matrices for D2Q9 and D3Q19 lattice structures used in our implementation. The ordering of the moments corresponding to the relaxation parameters in Eqs. (7) and (8) respectively is implicitly specified by the **T** matrix, where the column ordering corresponds to the ordering of the lattice velocity, and the row ordering corresponds to the ordering of the moments. For the D2Q9 lattice, the **T** matrix is:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -4 & -1 & -1 & -1 & -1 & 2 & 2 & 2 & 2 \\ 4 & -2 & -2 & -2 & -2 & 1 & 1 & 1 & 1 \\ 0 & 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & -2 & 2 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & -2 & 2 & 1 & -1 & 1 & -1 \\ 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \end{bmatrix},$$

and for the D3Q19 lattice, the **T** matrix is:

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-30 & -11 & -11 & -11 & -11 & -11 & -11 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\
12 & -4 & -4 & -4 & -4 & -4 & -4 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & -4 & 4 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\
0 & 0 & 0 & -4 & 4 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & -4 & 4 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
0 & 2 & 2 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -2 & -2 & -2 & -2 \\
0 & -4 & -4 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -2 & -2 & -2 & -2 \\
0 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -2 & -2 & 2 & 2 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1
\end{bmatrix}.$$

ACKNOWLEDGMENTS

The authors would like to thank Nils Thuerey for his kind discussions on the comparison with the FLIP method for liquid simulations, as well as the anonymous reviewers for many constructive comments. This work was supported by the Science and Technology Planning Major Project of Guangdong Province of China (Grant No. 2015A070711001) as well as the startup funding of ShanghaiTech University. X. Liu and X. Xu are the corresponding authors.

REFERENCES

[1] G. Irving, E. Guendelman, F. Losasso, and R. Fedkiw, "Efficient simulation of large bodies of water by coupling two and three dimensional techniques," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 805–811, Jul. 2006.

[2] H. Schechter and R. Bridson, "Ghost SPH for animating water," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 61:1–61:8, Jul. 2012.

[3] N. Foster and R. Fedkiw, "Practical animation of liquids," in *Proc. 28th Annu. Conf. Comput. Graph. Interactive Tech.*, New York, NY, USA: ACM, 2001, pp. 23–30.

[4] D. Enright, S. Marschner, and R. Fedkiw, "Animation and rendering of complex water surfaces," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 736–744, Jul. 2002.

[5] Y. Zhu and R. Bridson, "Animating sand as a fluid," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 965–972, Jul. 2005.

[6] J. Yu and G. Turk, "Reconstructing surfaces of particle-based fluids using anisotropic kernels," *ACM Trans. Graph.*, vol. 32, no. 1, pp. 5:1–5:12, Feb. 2013.

[7] J.-M. Hong, H.-Y. Lee, J.-C. Yoon, and C.-H. Kim, "Bubbles alive," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 48:1–48:4, Aug. 2008.

[8] L. Boyd and R. Bridson, "MultiFLIP for energetic two-phase fluid simulation," *ACM Trans. Graph.*, vol. 31, no. 2, pp. 16:1–16:12, Apr. 2012.

[9] N. Akinci, G. Akinci, and M. Teschner, "Versatile surface tension and adhesion for SPH fluids," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 182:1–182:8, Nov. 2013.

[10] N. Heo and H.-S. Ko, "Detail-preserving fully-Eulerian interface tracking framework," *ACM Trans. Graph.*, vol. 29, no. 6, pp. 176:1–176:8, Dec. 2010.

[11] M. Bojsen-Hansen and C. Wojtan, "Liquid surface tracking with error compensation," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 68:1–68:13, Jul. 2013.

[12] X. He, S. Chen, and R. Zhang, "A lattice Boltzmann scheme for incompressible multiphase flow and its application in simulation of rayleigh-taylor instability," *J. Comput. Phys.*, vol. 152, no. 2, pp. 642–663, 1999.

[13] M. E. McCracken and J. Abraham, "Multiple-relaxation-time lattice-Boltzmann model for multiphase flow," *Phys. Rev. E*, vol. 71, p. 036701, Mar. 2005.

[14] A. Fakhari and M. H. Rahimian, "Phase-field modeling by the method of lattice Boltzmann equations," *Phys. Rev. E*, vol. 81, p. 036707, Mar. 2010.

[15] J. Stam, "Stable fluids," in *Proc. 26th Annu. Conf. Comput. Graph. Interactive Tech.*, 1999, pp. 121–128.

[16] R. Fedkiw, J. Stam, and H. W. Jensen, "Visual simulation of smoke," in *Proc. 28th Annu. Conf. Comput. Graph. Interactive Tech.*, 2001, pp. 15–22.

[17] F. Losasso, F. Gibou, and R. Fedkiw, "Simulating water and smoke with an octree data structure," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 457–462, Aug. 2004.

[18] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, Aire-la-Ville, Switzerland: Eurographics Association, 2003, pp. 154–159.

[19] B. Solenthaler and R. Pajarola, "Predictive-corrective incompressible SPH," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 40:1–40:6, Jul. 2009.

[20] R. Ando, N. Thürey, and C. Wojtan, "Highly adaptive liquid simulations on tetrahedral meshes," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 103:1–103:10, Jul. 2013.

[21] M. Macklin and M. Müller, "Position based fluids," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 104:1–104:12, Jul. 2013.

[22] N. Thuerey and U. Ruede, "Free surface lattice-Boltzmann fluid simulations with and without level sets," *in Proc. Vis., Modelling, Vis.*, pp. 199–207, 2004.

[23] J.-M. Hong and C.-H. Kim, "Discontinuous fluids," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 915–920, Jul. 2005.

[24] F. Losasso, T. Shinar, A. Selle, and R. Fedkiw, "Multiple interacting liquids," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 812–819, Jul. 2006.

[25] B. Kim, "Multi-phase fluid simulations using regional level sets," *ACM Trans. Graph.*, vol. 29, no. 6, pp. 175:1–175:8, Dec. 2010.

[26] J. Cho and H.-S. Ko, "Geometry-aware volume-of-fluid method," *Comput. Graph. Forum*, vol. 32, no. 2.3, pp. 379–388, 2013.

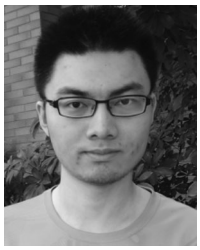
[27] B. Solenthaler and R. Pajarola, "Density contrast SPH interfaces," in *Proc. Eurographics/SIGGRAPH Symp. Comput. Animation*, 2008, pp. 211–218.

[28] C. K. Aidun and J. R. Clausen, "Lattice-Boltzmann method for complex flows," *Annu. Rev. Fluid Mech.*, vol. 42, no. 1, pp. 439–472, 2010.

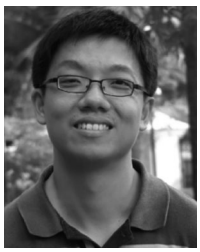
[29] P. Yuan and L. Schaefer, "Equations of state in a lattice Boltzmann model," *Phys. Fluids*, vol. 18, no. 4, p. 042101, 2006.

[30] Q. Li, K. H. Luo, and X. J. Li, "Lattice Boltzmann modeling of multiphase flows at large density ratio with an improved pseudopotential model," *Phys. Rev. E*, vol. 87, p. 053301, May. 2013.

[31] S. Leclaire, M. Reggio, and J.-Y. Trpanier, "Isotropic color gradient for simulating very high-density ratios with a two-phase flow lattice Boltzmann model," *Comput. Fluids*, vol. 48, no. 1, pp. 98–112, 2011.
[32] S. Leclaire, M. Reggio, and J.-Y. Trepanier, "Progress and investigation on lattice Boltzmann modeling of multiple immiscible fluids or components with variable density and viscosity ratios," *J. Comput. Phys.*, vol. 246, pp. 318–342, 2013.
[33] T. Lee and C.-L. Lin, "A stable discretization of the lattice Boltzmann equation for simulation of incompressible two-phase flows at high density ratio," *J. Comput. Phys.*, vol. 206, no. 1, pp. 16–47, 2005.
[34] J. Y. Shao, C. Shu, H. B. Huang, and Y. T. Chew, "Free-energy-based lattice Boltzmann model for the simulation of multiphase flows with density contrast," *Phys. Rev. E*, vol. 89, p. 033309, Mar. 2014.
[35] H. Zhu, X. Liu, Y. Liu, and E. Wu, "Simulation of miscible binary mixtures based on lattice Boltzmann method," *Comput. Animation Virtual Worlds*, vol. 17, no. 3-4, pp. 403–410, 2006.
[36] H. Zhu, K. Bao, E. Wu, and X. Liu, "Stable and efficient miscible liquid-liquid interactions," in *Proc. ACM Symp. Virtual Reality Softw. Tech.*, New York, NY, USA: ACM, 2007, pp. 55–64.
[37] J. Park, Y. Kim, D. Wi, N. Kang, S. Y. Shin, and J. Noh, "A unified handling of immiscible and miscible fluids," *Comput. Animation Virtual Worlds*, vol. 19, no. 3-4, pp. 455–467, Sep. 2008.
[38] L.-S. Luo and S. S. Girimaji, "Theory of the lattice Boltzmann method: Two-fluid model for binary mixtures," *Phys. Rev. E*, vol. 67, p. 036302, Mar. 2003.
[39] L. Pierre and L.-S. Luo, "Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, galilean invariance, and stability," *Phys. Rev. E*, vol. 61, no. 6, pp. 6546–6562, 2000.
[40] D. D'Humieres, G. Irina, K. Manfred, L. Pierre, and L.-S. Luo, "Multiple-relaxation-time lattice Boltzmann models in three dimensions," *Philosoph. Trans. Roy. Soc. London Series A-Math. Phys. Eng. Sci.*, vol. 360, no. 1792, pp. 437–451, 2002.
[41] A. Fakhari and T. Lee, "Multiple-relaxation-time lattice Boltzmann method for immiscible fluids at high Reynolds numbers," *Phys. Rev. E*, vol. 87, p. 023304, Feb. 2013.
[42] N. Thuerey, U. Ruede, and M. Stamminger, "Animation of open water phenomena with coupled shallow water and free surface simulation," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, Jun. 2006, pp. 157–166.
[43] X. Liu, W.-M. Pang, J. Qin, and C.-W. Fu, "Turbulence simulation by adaptive multi-relaxation lattice Boltzmann modeling," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 2, pp. 289–302, Feb. 2014.
[44] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot, "A dynamic subgrid-scale eddy viscosity model," *Phys. Fluids A, Fluid Dyn.*, vol. 3, no. 7, pp. 1760–1765, 1991.
[45] K. N. Premnath, M. J. Pattison, and S. Banerjee, "Dynamic subgrid scale modeling of turbulent flows using lattice-Boltzmann method," *Phys. A, Statist. Mech. Appl.*, vol. 388, no. 13, pp. 2640–2658, 2009.
[46] D. K. Lilly, "A proposed modification of the Germano subgrid-scale closure method," *Phys. Fluids A, Fluid Dyn.*, vol. 4, no. 3, pp. 633–635, 1992.
[47] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *in Proc. 14th Annu. Conf. Comput. Graph. Interactive Tech.*, Aug. 1987, pp. 163–169.
[48] M. Pharr and G. Humphreys, *Physically Based Rendering, Second Edition: From Theory To Implementation*, 2nd ed. San Francisco, CA, USA: Morgan Kaufmann, 2010.
[49] T. Zhang, B. Shi, Z. Guo, Z. Chai, and J. Lu, "General bounce-back scheme for concentration boundary condition in the lattice-Boltzmann method," *Phys. Rev. E*, vol. 85, p. 016701, Jan. 2012.
[50] T. Pfaff and N. Thuerey. (2015). Mantaflow [Online]. Available: http://mantaflow.com.
[51] R. Ando, N. Thuerey, and C. Wojtan, "A stream function solver for liquid simulations," *Trans. Graph.*, vol. 34, no. 2, p. 8, Aug. 2015.



Yulong Guo received the BS degree in electronic engineering in 2012, and the Master's degree majoring in computer science from the School of Computer Science and Engineering, South China University of Technology, Guangdong, China, in 2015. He is currently a research assistant at the School of Information Science and Technology, ShanghaiTech University, Shanghai, China. His current research interests include numerical simulation of fluid flows, scientific visualization, as well as physically based volume rendering.



Xiaopei Liu received the PhD degree in computer science and engineering from the Chinese University of Hong Kong, Hong Kong, in 2010. He is currently an assistant professor at the School of Information Science and Technology, ShanghaiTech University, Shanghai, China. Previously, he was a research fellow at Nanyang Technological University in Singapore. His current research interests include fluid computing and analysis, flow visualization and rendering techniques, as well as GPU-based parallel computing. He is also interested in computational methods for image analysis, synthesis, and computer-vision related topics, as well as the design and implementation of small-scale unmanned aerial vehicles.

Xuemiao Xu received the PhD degree in computer science and engineering from the Chinese University of Hong Kong, Hong Kong, in 2009. She is currently a professor in the School of Computer Science and Engineering, South China University of Technology, Guangdong, China. Her research interests include image similarity measurement, digital manga/cartoon, intelligent transportation based on image analysis, and biometric recognition.