Available online at www.sciencedirect.com
SCIENCE DIRECT
Computers & Graphics 30 (2006) 519–528
COMPUTERS & GRAPHICS
www.elsevier.com/locate/cag

# Melting and flowing in multiphase environment

Ye Zhao*, Lujin Wang, Feng Qiu, Arie Kaufman, Klaus Mueller

*Center for Visual Computing and Department of Computer Science, Stony Brook University, Stony Brook, NY 11794-4400, USA*

**Abstract**

This paper presents a method to simulate the melting and flowing phenomena with different materials in multiple phases. In such a multiphase environment, solid objects are melted because of heating and the melted liquid flows while interacting with the ambient air flow. Our simulation is based on a modified lattice Boltzmann method (LBM), where the fluid dynamics of the air flow and the melted liquid is modeled within a common lattice framework. Therefore, no particular front tracking methods are required for the liquid–air interface. The liquid–solid and air–solid interfaces are implemented as curved boundaries in the LBM, which can accommodate arbitrarily shaped solid objects. Heat transfer is incorporated with a finite difference discretization of a standard diffusion–advection equation simulating the temperature evolution. The temperature and body forces (gravity and surface tension) are easily applied by adopting a new version of the LBM: multiple-relaxation-time LBM (MRTLBM). The melting and flowing behavior is controlled by the heat source, the viscosity and the body forces. All the numerical computations in our method are local and parallelizable, therefore, interactive speed is achieved by hardware acceleration on the contemporary graphics hardware (GPU).


*Keywords:* Melting; Fluid dynamics; Lattice Boltzmann method; Multiphase; Heat transfer

## 1. Introduction

Modeling and simulating the natural phenomena involving materials in multiple phases, such as melting, freezing, thawing, boiling and evaporation, has been an interesting topic for computer graphics researchers. However, these phenomena involve phase transition, heat exchange, fluid dynamics and rigid body deformation, which makes them a great challenge for computer simulation. Some researches have been conducted on this topic in computer graphics [1–4]. In this paper, we present a straightforward and parallelizable framework for modeling the melting and flowing in a multiphase environment incorporating solid, melted liquid and surrounding air. Our main contributions are: (1) based on a multiphase lattice Boltzmann method (LBM) [5], our method provides a simple and fast computational framework for the simulation of complex flow behaviors with heat effects, which is easily accelerated on contemporary graphics hardware (GPU); (2) our method includes solid, liquid and air in a uniform lattice structure and simulates their dynamics in one common microscopic updating scheme. That is, the flowing behavior of the melted liquid is influenced by the air flows. In comparison, most previous work excludes the effects from the surrounding air by using particular free-surface modeling methods.

One of the important tasks and challenges in modeling dynamic behaviors inside a multiphase environment is to capture the interfaces between different materials at each numerical step. Traditional front tracking methods, such as level set methods, require interface reconstruction in order to determine the curvature of the interface and in turn the influence of surface tension on the fluid momentum. Accurate interface reconstruction is preferred. For instance, good visual results are achieved in free-surface water modeling [6] by using a particle level set method. However, the reconstruction procedure consumes large computation resources. The LBM offers a valuable alternative that can deal with the dynamic interfaces between multiphase materials in a simple way with its local and parallel computations.

The LBM constructs simplified kinetic models that incorporate the essential physics of microscopic processes

*Corresponding author. Tel.: +1 631 632 8428.
*E-mail addresses:* yezhao@cs.sunysb.edu (Y. Zhao), lujin@cs.sunysb.edu (L. Wang), qfeng@cs.sunysb.edu (F. Qiu), ari@cs.sunysb.edu (A. Kaufman), mueller@cs.sunysb.edu (K. Mueller).

0097-8493/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2006.03.009

so that the macroscopic averaged properties obey the desired macroscopic Navier–Stokes (NS) equations for fluid dynamics. Since the LBM proceeds at a microscopic scale, it is able to model the multiphase flows including liquid and gas with various viscosities and complex boundaries by its local operators. Furthermore, the multiphase flows are simulated in a uniform computational framework. Therefore, our method incorporates the air effects and avoids the extrapolation operations (e.g., [6]) required by surface reconstruction.

In our method, the simulating space is discretized into a lattice structure in which each cell has its specific properties, including density, velocity, temperature, phase mode, etc. The interfaces between different phases evolve inside the collision-streaming scheme of the LBM without any explicit front reconstruction. The phase transition from solid to liquid is modeled by changing the phase mode of a cell according to the temperature variation. Meanwhile, the liquid–gas transition is determined by the amount of liquid mass at the cell. In our LBM implementation, the microscopic particle propagation operators are specially designed with respect to the interfaces, while obeying the rules of Boltzmann dynamics for fluids. The macroscopic behaviors are controlled by heat sources, viscosities and other physical properties.

In the direct numerical simulation of the incompressible NS equations, the pressure satisfies a Poisson equation with velocity strains acting as sources. This equation requires iteratively solving a large linear system. In contrast, as an explicit solver for NS equations, the LBM operators are local and especially suitable for acceleration on the GPU to achieve good performance.

Heat transport is clearly an important factor behind the melting phenomena. A standard diffusion–advection equation governing the temperature evolution is incorporated into the LBM framework. This equation models the heat propagation in solid, liquid and air with different parameters. The body forces, such as gravity and surface tension, are naturally included in our LBM structure and play an important role in the flowing behavior. We adopt a new version of the LBM, multiple-relaxation-time LBM (MRTLBM), which easily accommodates the temperature and the body forces, as well as improves the stability of our numerical computation.

The remainder of the paper is organized as follows. In the next section, we give a brief review of the related work. We then describe the MRTLBM framework in Section 3. In Section 4, the heat transfer and the body forces are introduced and added to the MRTLBM. In Section 5, the methods of handling the multiphase interfaces and the modified LBM rules are proposed. Finally, we describe our simulation results and performance.

## 2. Related work

A variety of physics-based approaches have been applied to model fluid phenomena. In particular, the application of CFD methods for solving the NS equations has led to a significant advance in the visualization of gas, fire, and fluids. In particular, Foster and Metaxas [7] developed a method for the realistic animation of fluids using velocity and pressure fields to drive the liquid surface, which was represented by a height field. Later, they presented a full 3D finite difference solution of the NS equations to simulate the turbulent rotational motion of a gas [8] driven by thermal buoyancy. Stam [9] devised a fluid solver using semi-Lagrangian advection schemes for the NS equations. This method is able to achieve real-time speed on a low-resolution grid without time step restrictions. Vortex confinement was applied to feed energy back into the vortices [10] for smoke modeling and to couple the vortex particles into the grid-based computation [11]. Using the level set method to track the moving flame surface and water surface, realistic looking turbulent flames and water were generated [6,12]. Particle based methods [3,13,14] have also been applied to model fluid with interactive simulation and easy user-control.

The LBM is an alternative microscopic solver for the NS equations, which can be proved by Chapman–Enskog analysis [15]. The LBM has achieved success in the physics and computational science world both in the analytical and practical points of view. It has also been successfully used in graphics for simulating a variety of fluid phenomena with complex boundary conditions [16–18]. These previous efforts are based on a single-relaxation-time LBM (SRTLBM). The MRTLBM [19] is a new approach of LBM providing increased numerical stability and easy tuning of the viscosity and other fluid properties. We have applied it to simulate dispersions in urban environment [20]. In this previous work, however, we did not incorporate temperature effects directly into the LBM. McCracken et al. [21] used the index function and the MRTLBM to model two-phase flows in 2D, which shows that the MRTLBM can provide more accurate and stable simulations than the SRTLBM.

Simulating and visualizing the melting, flowing, and solidification of materials presents interesting challenges for researchers. Terzopoulos et al. [22] implemented a hybrid mesh-particle method for heating and melting deformable models. Carlson et al. [1] used a Marker-and-Cell algorithm for NS equation to simulate the fluids with the material viscosity represented as either a linear or quadratic function of the temperature. Wei et al. [4] presented a simple, linear 3D cellular automaton for animating the melting of solid volumetric models. In their model, the convected temperature was used to set the status of a voxel as either fluid or solid. Müller et al. [2] presented a method for modeling and animating a wide spectrum of volumetric objects with material properties in the range from stiff elastic to highly plastic, including melting behavior. Rasmussen et al. [3] provided direct controls for liquid behavior in order to facilitate the generation of melting effects in movies. Goktekin et al. [23] described a technique for animating the behavior of viscoelastic fluids

that exhibit a combination of both fluid and solid characteristics. Zhu and Bridson [24] presented a fluid simulator for animating sand and created the melting-like behavior of the sand by reconstructing a surface from particles.

Our method simulates melting and flowing by utilizing the ability of the LBM for modeling multiphase or multi-component flows. Level set methods have also been used for two-phase flow [25]. Free surface fluid simulation is a simplified multiphase scenario, where the surrounding air is not modeled and simplified as an empty space. Enright et al. [6] have addressed this issue with a particle level set method, which has also been used by Rasmussen et al. [3]. Thürey and Rüde [17] presented free surface simulations based on the LBM for animation of liquids with and without level sets. Zhu and Bridson [24] utilized a fully particle-based reconstruction for generating surfaces. Our method differs from theirs in that the liquid interacts with the surrounding air and no particular front tracking method outside of the LBM framework is required.

## 3. Multiple-relaxation-time LBM

The LBM is a parallel and efficient algorithm for simulating flows and incorporating additional physical complexities. The fundamental idea of the LBM is to model fluid dynamics based on the collective behavior of many microscopic particles. The kinetic equation provides many advantages of molecular dynamics necessary for the modeling of fluids, including easy implementation of boundary conditions and fully parallel algorithms. The NS equations can be derived in the nearly incompressible limit of the LBM, which can be shown by a Chapman–Enskog expansion [15]. We would like to refer the readers to the book [5] for complete description and discussion of the LBM.

Originated from lattice gas automata, the LBM models Boltzmann particle dynamics on a 3D lattice. The D3Q13 lattice (3 dimensions, 13 links) is illustrated in Fig. 1. The variables associated with each lattice site are the particle distributions $f_i$ that represent the probability of the presence of a fluid particle with a given velocity vector $\mathbf{e}_i$, where $i \in 0 \ldots 12$ represents one of the 13 discrete directions. Particles stream synchronously along the lattice links to their neighbors in discrete time steps. Between streaming steps, they undergo collision, which occurs when particles arriving at a node interact and change their velocity directions according to the scattering rules. In these scattering rules, the macroscopic properties (e.g., body forces) affect the microscopic particle dynamics. The primary LBM model represents Boltzmann dynamics as a discrete kinetic equation with a two-step process of collision and ballistic streaming:

$$f_i(\mathbf{r}, t^*) = f_i(\mathbf{r}, t) - \tfrac{1}{\tau}(f_i(\mathbf{r}, t) - f_i^{eq}(\rho, \mathbf{u})) \Rightarrow \text{ collision}, \quad (1)$$

$$f_i(\mathbf{r} + \mathbf{e}_i, t + 1) = f_i(\mathbf{r}, t^*) \Rightarrow \text{ streaming}, \quad (2)$$

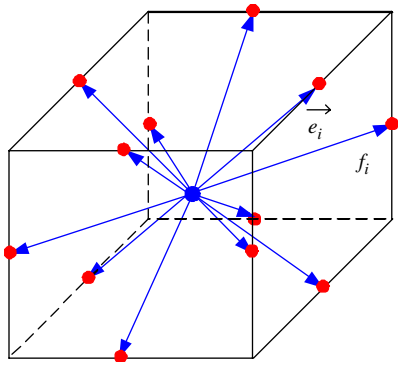


Fig. 1. The D3Q13 lattice geometry. The particle distribution $f_i$ is associated with the link corresponding to the $\mathbf{e}_i$ velocity vector from a lattice site (blue) to its neighbors (red).

where $\mathbf{r}$ and $\mathbf{r} + \mathbf{e}_i$ locate a lattice node and its neighboring node along link $i$, respectively. The distribution denoted as $f_i^{eq}$ represents a local equilibrium distribution, whose value is usually defined as a linear function (BGK model [5]) of the conserved quantities: mass $\rho = \sum_i f_i$ and momentum $\mathbf{p} = \rho\mathbf{u} = \sum_i f_i \mathbf{e}_i$. The constant $\tau$ represents the relaxation time scale that determines the viscosity of the flow. For the kinematic shear viscosity $v$, $\tau$ is defined as $\tau = (6v + 1)/2$. For a turbulent fluid that has low viscosity, the values of $v$ and $\tau$ are smaller than for a laminar fluid. Therefore, in simulation one can observe more vortices for such fluid.

Only one parameter, $\tau$, is used to control the fluid behavior in Eqs. (1) and (2). Therefore, this primary LBM is called single-relaxation-time LBM (SRTLBM). The SRTLBM is prone to unstable numerical computation when used for low viscosity fluids (high turbulent fluids) or incorporated with temperatures or body forces. Therefore, we use a newer version of the LBM, multiple-relaxation-time LBM (MRTLBM) [19], as the multiphase framework of our melting and flowing simulation. It differs from the SRTLBM in that a new collision operator operates the single relaxation approach in Eq. (1). The new collision operates in the space of hydrodynamic moments representing density, momentum, energy, etc. These moments, denoted as $m_i$, are mapped from the particle distributions as

$$|m\rangle = M|f\rangle, \quad (3)$$

where $|f\rangle = (f_0, f_1, \ldots, f_n)^{\mathrm{T}}$, $|m\rangle = (m_0, m_1, \ldots, m_n)^{\mathrm{T}}$ and $n$ is the number of lattice links of a node to its neighbors. For the D3Q13 lattice, each of the 13 moments has a physical meaning: $m_0$ is the mass density $\rho$, $m_{1,2,3}$ are the components of the momentum vector $\mathbf{p}$, $m_4$ is the energy, and the other higher order moments are components of the stress tensor and other high order tensors. Although the values of the distributions and the moments vary over the nodes of the lattice, the matrix $M$ is simply constant for a given lattice structure. For example, since $\rho = \sum_i f_i$, the first row of $M$ consists of all ones. Mathematically, the collision of the MRTLBM is described as

$$|f(\mathbf{r}, t^*)\rangle = |f(\mathbf{r}, t)\rangle - M^{-1} S[|m(\mathbf{r}, t)\rangle - |m^{eq}(\mathbf{r}, t)\rangle]. \quad (4)$$

Note that Eq. (1) can be seen as a special case of Eq. (4), where the relaxation matrix $S$ is a diagonal matrix with all the diagonal elements have only one value, $1/\tau$. In the MRTLBM, $S$ is also a diagonal matrix, where the diagonal elements $\{s_i | i = 0, 1, \ldots, n\}$ are different and have their own physical meanings. Using the D3Q13 lattice in Fig. 1 as an example (each cell includes the center cell with zero velocity and the 12 minor-diagonal neighboring links), the kinematic shear and bulk viscosities, $v$ and $\xi$, define $s_5$ and $s_6$ as

$$s_5 = \frac{1}{\xi/(\frac{2}{3} - \gamma c_{s0}^2) + \frac{1}{2}}, \tag{5}$$

$$s_6 = \frac{1}{2v + \frac{1}{2}}, \tag{6}$$

where $\gamma$ is the specific heat, $c_{s0}$ is the isothermal speed of sound and they are constant for a particular material.

By setting parameters in $S$, users have the freedom to choose the flow parameters to define characteristics of the fluid being modeled. For example, the viscosities can be easily chosen to control the fluid behavior for different materials, and this choice then determines the relaxation rates as in Eqs. (5) and (6). Moreover, it has been proven [26] that the MRTLBM increases the numerical stability. More details of the MRTLBM of the values of the constants and matrices can be found in [26]. In our simulation, the MRTLBM provides a powerful framework where body forces and heat effects can be easily added to the moments $m_i$, because the moments have explicit physical meanings, such as momentum and energy.

## 4. Heat transfer and body forces

Heat transfers in the simulation space and melts solid materials to liquid, triggering the flow of the liquid. The temperature evolution is governed by a diffusion–advection equation,

$$\partial_t T + \mathbf{u} \cdot \nabla T = \kappa \Delta T, \tag{7}$$

where $\kappa$ is the thermal diffusivity of the material and $\mathbf{u}$ is the velocity. This equation is solved by finite-difference operators. The stencil of these operators are defined in the same symmetric structure as the LBM lattice to keep the system stable when combined with the MRTLBM [26]. For a D3Q13 LBM structure, the $x$ component of the gradient $\nabla T$ is computed by the finite difference operator as

$$\begin{aligned}\hat{\nabla} T(i,j,k) = \tfrac{1}{8}[&T(i+1,j+1,k) - T(i-1,j+1,k) \\ &+ T(i+1,j-1,k) - T(i-1,j-1,k) \\ &+ T(i+1,j,k+1) - T(i-1,j,k+1) \\ &+ T(i+1,j,k-1) - T(i-1,j,k-1)].\end{aligned}$$

For the $y$ and $z$ components, the $\nabla T$ is defined in the same manner. The finite difference Laplacian operator $\Delta T$ are computed as

$$\begin{aligned}\hat{\Delta} T(i,j,k) = \tfrac{1}{4}[&T(i+1,j+1,k) + T(i-1,j+1,k) \\ &+ T(i+1,j-1,k) + T(i-1,j-1,k) \\ &+ T(i+1,j,k+1) + T(i-1,j,k+1) \\ &+ T(i+1,j+1,k-1) + T(i,j-1,k-1) \\ &+ T(i,j+1,k+1) + T(i,j-1,k+1) \\ &+ T(i,j,k-1) + T(i-1,j,k-1) \\ &- 3T(i,j,k)].\end{aligned}$$

The temperature is coupled to the MRTLBM in order to model the interaction between the heat and the flow dynamics. The moment $m_4$ represents the energy, therefore, the temperature can be added to the MRTLBM when computing the equilibrium $m_4^{eq}$:

$$m_4^{eq} = n_1 \rho + n_2 \mathbf{p} \cdot \mathbf{p} + n_3 T, \tag{8}$$

where the parameters, $n_1$ to $n_3$, are constants and physically defined by the linear analysis. After coupling $T$ to $m_4^{eq}$, our method could model the buoyancy flow generated solely by a heat source.

The temperature of each cell defines the viscosity of the fluid material. In the LBM, each cell has its own viscosity value. We set the viscosity as a function of the temperature and then define the relaxation times of the LBM, as described in Eqs. (5) and (6). In comparison, the implicit NS solvers [9] take the constant viscosity effect in a diffusion term. When solving variable viscosity [1], the viscosities between pairs of adjacent cells are averaged to modify the diffusion solver. This is demanded due to its macroscopic feature for solving the Laplacian term. On the other hand, the LBM implements the viscosity variation simply and naturally because its microscopic operations avoid solving the gradient and Laplacian terms with finite difference operators.

The body forces play important roles in our melting modeling. The gravity $\mathbf{F_g}$ makes the melted liquid flow downwards and the surface tension $\mathbf{F_s}$ gives the fluid realistic behavior and appearance. In our implementation, besides the gravity, the surface tension is defined as

$$\mathbf{F_s} = \lambda \rho \nabla \nabla^2 \rho, \tag{9}$$

where $\rho$ is the density and $\lambda$ defines the magnitude of the force. For the MRTLBM, the moments $m_{1,2,3}$ are the components of the momentum vector $\mathbf{p}$ of each cell. Therefore, the body force $\mathbf{F} = \mathbf{F_g} + \mathbf{F_s}$ is added to the equilibrium $m_{1,2,3}^{eq}$:

$$\mathbf{p}^{new} = \mathbf{p} + \mathbf{F} \delta t, \tag{10}$$

where $\delta t$ is the time interval and the value is set to 1 for each time step in the LBM. In previous work [10], the temperature was usually applied to the flow dynamics as a buoyancy force based on the Boussinesq approximation (i.e., the density variation only appears in the force term), and the coupling parameters are usually chosen manually. Besides applying the temperature to the MRTLBM as an

energy term, we also implement adding the temperature as a buoyancy force, which generates similar results.

Solving the temperature equation and applying the body forces are based on local operators that only require the neighboring data. This allows them to be parallelized along with the MRTLBM. Therefore, all the numerical computations of our method, including the LBM and the heat transfer, are accelerated on the GPU at the same time. We refer the readers to our previous work [27] for more information on the acceleration of the LBM on the GPU.

## 5. Multiphase interfaces

Our melting simulation accommodates solid objects, melted liquid, as well as the air surrounding them, in one lattice structure with each cell having a dynamic phase mode. An object can be of arbitrary shape and it affects the flow behavior on solid–fluid interfaces as curved boundaries of the LBM. Our method has the advantage that it includes the dynamics of multiphase fluids (liquid and air) in one common computational structure and it does not need to explicitly track the free surfaces. However, this requires the deliberate adjustment of the LBM updating rules. Next, we describe two kinds of interfaces, solid–fluid and liquid–air, respectively.

### 5.1. Solid objects

In the melting and flowing environment, the solid objects can be categorized into two groups. In the first group, the solid objects are not meltable and keep their geometries constant in the simulation. In the second, the objects are melting because of the heating. In any given time, some parts are still solid and the others have been transformed into melted liquids.

#### 5.1.1. Non-meltable objects

Interactions between an LBM flow field and an immersed non-meltable object result from the exchange of momentum at their shared boundaries. The treatment of boundary conditions of the MRTLBM are handled in the discrete velocity space after a general streaming simulating step (Eq. (2)). Generally, there exist two types of boundaries in the LBM: (1) the surrounding wall of the LBM simulation space; (2) internal objects. These have been discussed in our previous work [18]. In our melting simulation, we implemented periodic, out-flow and bounce-back conditions [5] for the walls, as well as an improved bounce-back rule for curved, moving no-slip boundaries [28] for the internal objects. In summary, for the non-meltable objects, we take advantage of the LBM features that it can efficiently deal with complex objects with arbitrary geometries.

#### 5.1.2. Melting objects

The melting solids are the other group of objects inside the simulation space. They melt due to the heat energy which changesthe phase mode of the occupied cells from solid to liquid. When the temperature of a solid cell exceeds a threshold, the mode of the cell is transformed to liquid. The temperature distribution is computed by solving Eq. (7). In our simulation, we model the heat transfer not only in the liquid and air, but also within the solids using the same diffusion–advection equation. The diffusivity of the solid, $\kappa$, is lower than that of the liquid and the gas flows, and the thermal advection velocity for the solid is set to zero. In other words, the interior of the melting object only supports temperature diffusion. Therefore, by using the same thermal model with different parameters, we are able to evaluate the temperature dispersion uniformly over the whole 3D simulating space for both the solids and the fluids.

During melting, the unmelted solid cells act as rigid objects inside the flows. They play a role of boundaries of the LBM. However, we do not generate the mesh at each step because their dynamic mode variation would make this too expensive. Instead, we directly modify the particle distributions propagating between such a solid cell and its neighboring fluid cells. The LBM streams the particles along a link to its neighbor. As illustrated in Fig. 2, along a link direction $i$ a solid cell $S$ has a neighboring cell $F$ that is a liquid or a gas cell. While the streaming operation does not apply to $S$, in the reverse direction $ri$ of the current direction $i$, the particle distribution $f_{ri}$ propagating from $S$ to $F$ is required for the correct computation of $F$. We apply the no-slip boundary condition here, which is implemented by a simple bounce-back rule [5]. Following this rule, all particle distributions sitting on a boundary node are reversed. In Fig. 2, the reversed particle distribution of $F$ is straightforwardly computed as: $f_{ri} = f_i$. Between two unmelted solid cells, we do not need to apply the LBM rules and simply leave their original status. This differs from the method of Carlson et al. [1] where the unmelted body is implemented with a very high viscosity.

### 5.2. Liquid and gas

An LBM cell is set to liquid or gas phase mode for the melted liquid and the ambient air, respectively. Both the liquid and gas cells follow the LBM updating rules of collision and streaming. Liquid and gas cells are treated

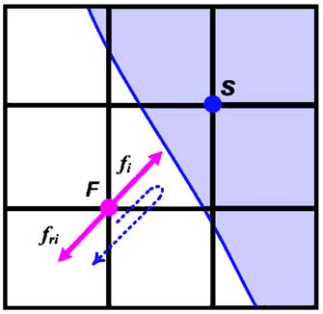


Fig. 2. Bounce back rule for solid–fluid interface.

differently in the LBM streaming–collision implementation. Besides the different temperature transfer behaviors, different viscosities are defined and implemented with different relaxation matrices (Eqs. (5) and (6)). In order to achieve correct flow behavior, the update of particle distributions on the interface of liquid and gas cells has to be modified deliberately. This interface is usually handled as a free surface (i.e., no surrounding air flow is considered) in previous work. Instead, our method provides a solution involving both air and liquid together without direct surface tracking. Moreover, the air flow affects the heat transfer process and then the flowing behavior.

#### 5.2.1. Modified LBM rules

We illustrate our modified LBM rules at the liquid–gas(air) cell interface by Fig. 3. At the beginning, a liquid cell $L$ and a gas cell $G$ have their particle distributions $l$ and $g$, respectively. During melting, the liquid pushes the interface between it and the gas cell forward towards the gas cell. When using the conventional streaming operator (Eq. (2)), $g$ streams to $L$ and the unmodified streaming results are shown in Fig. 3(b). This is not correct because air cannot stream into liquid body (if modeling boiling and vaporizing this rule is correct due to the real mass exchange between steam and water [29]). Therefore, we modify the streaming rule: replace the particle distribution streamed from $G$ to $L$ by $l^*$. To model that the gas is passively pushed by the liquid, $l^*$ equals the equilibrium particle distribution $l^{eq}$ of $L$ at that direction (see Fig. 3(c)). This affirms that the liquid flows into the gas space but the gas cannot stream into the liquid body. $l^*$ can also be scaled by taking into account the intensity of the gas velocity, which models the blowing effect on the liquid.

Globally, the proper flowing of the liquid is achieved by a driving body force $D$ (such as gravity) applied to the liquid cell, which makes $l^*$ smaller in directions opposite to the force and larger in the other directions, as shown in Fig. 3(d). As a result, the force drags the liquid to flow along the force direction. Therefore, the key point of modeling proper melting and flowing is to correctly introduce the body forces. We add the gravity and the surface tension to the MRTLBM as described in Section 4. Note that we explain our method with the figures only considering two neighbors. Actually, the flow behavior and the force effects are the combined results from all the valid neighbors for a particular cell.

In summary, by modifying the LBM rules between cells with different phases, our method provides a fast and easily implemented framework to simulate melting and flowing involving surrounding air flow and heat distribution. Small modifications are applied to the conventional LBM streaming rules on the interfaces in the same manner as for general boundary treatment. Consequently, without losing the simplicity and elegance of the LBM operations, it yields visually realistic results with meaningful parameters (viscosities, body forces, etc.) to control fluid behaviors.

#### 5.2.2. Mass exchange

To model the flowing of the melted liquid, each liquid cell is assigned a mass when transformed from the solid. The mass spreads from a liquid cell to a gas cell, which will change the latter to a liquid cell. On the other hand, if the mass of a liquid cell streams totally to its neighbors, the cell will be occupied by the air and its mode changes to a gas cell. At each simulation step, we update the phase modes of each cell by comparing the mass value with a threshold $M_{Thres}$ and reset the phase mode accordingly.

The liquid mass of every cell constructs together a scalar field in the simulation space (for the solid and air cells, it is set to zero). The evolution of the scalar field has been modeled as an advection–diffusion equation by Stam [9]. We apply the same idea to move the mass by a back-tracing algorithm based on the method of characteristics (see Appendix A in [9]). Because the flow behavior is modeled in the LBM by our modified rules as described above, the back-tracing method provides the correct mass flowing while using the velocity field generated by the LBM. The mass distribution in the simulation grid can be initialized from various models. In addition to the geometric meshes, volumetric datasets provide non-homogeneous initial status to define interesting melting behaviors.

We generate the melting results by extracting geometric meshes from the scalar mass field using the Marching Cubes method [30]. Straightforwardly, the mass value is computed on the lattice sites of the LBM. However, because the LBM lattice is typically too coarse to generate

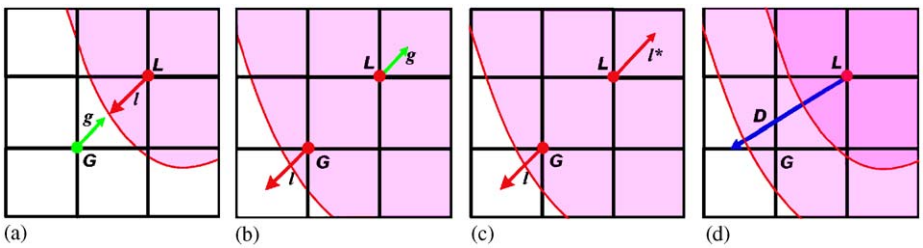


Fig. 3. Liquid (pink) and gas (white) interface. (a) Particle distributions between gas cell $G$ and liquid cell $L$; (b) unmodified streaming results: gas particles flow into liquid improperly; (c) modified streaming results: gas particles are replaced by liquid particles; (d) liquid mass is dragged forward by a body force $D$.

smooth meshes, we want to increase the resolution of the mass field to improve the accuracy. Therefore, we subdivide an LBM cell into smaller cells. Each small cell uses the back-tracing method to modify its mass value, and the velocity of these cells is computed by interpolation from the coarse velocity field. This approach is better than directly generating a high-resolution mass volume by linear interpolation. Such a solution is inspired by the particle level set method [6] that traces marker particles each step to adjust the evolving interface on a coarse velocity field. As this semi-Lagrangian approach, our method generates more accurate and smoother geometric meshes. At the same time, the LBM Eulerian computing structure is unchanged so that we can still accelerate the computation on the GPU. We further improve the performance by only subdividing the LBM cells on neighboring regions of the current liquid–air interface at each simulation step.

## 6. Discussion

We include fluid dynamics of both the melted liquid and the surrounding air in the LBM computation. Each cell in our simulating space has its density, velocity, viscosity, temperature and other physical properties. Therefore, we do not need to perform the interface reconstruction at each step with the extrapolations of the velocity field as in the free surface modeling approaches. Our interface evolution is implicitly implemented by the modified LBM rules, which only add a minimum overhead to the whole LBM scheme.

Currently, our simulation models non-meltable objects and unmelted solids as fixed boundaries of the LBM. In the near future, we would like to model the solid mechanics (e.g., the movement and deformation) of these objects together with the fluid dynamics. This is one of the main challenges for physically based modeling in computer graphics. The solution of this problem will create more realistic simulation results, such as the melting solids bending, falling or floating on the melted liquids.

As a lattice-based method, our approach suffers from the same problem as other Eulerian methods: the accuracy of the simulation depends on the size of the simulation grid. The LBM does not iteratively solve a linear system from the Poisson equation of the pressure, which is required by implicit fluid solvers. Thus, it can be straightforwardly accelerated on SIMD (single instruction stream, multiple data stream) processors, such as the GPU. While this does increase the ability of the LBM to accommodate large grid size, the LBM is an explicit solver and requires relatively small time steps for stability. While the MRTLBM greatly improves the stability over previously used SRTLBM, some multi-resolution schemes and adaptive time step schemes [31] have also been applied to the LBM to address this issue.

## 7. Results

We have applied our multiphase LBM-based melting method to several examples. For the CPU version, our simulation is timed on one 3.0 GHz Pentium Xeon CPU with 1 GB memory. We also accelerate the LBM simulation on a GPU (nVidia GeForce 6800 Ultra). For a simulation lattice size of $64^3$ that we used for examples, each computation step (i.e., each rendering frame) costs an average of 2484 ms on the CPU and 183 ms on the GPU. The acceleration on the GPU thus achieves a speedup factor of 13.5 over the CPU implementations.

From the mass field of liquids, we generate geometric meshes of the liquid–gas interface by the Marching Cubes algorithm. Note that the method we described in Section 5.2.2 provides us the flexibility to create higher resolution mass distribution than the LBM computation

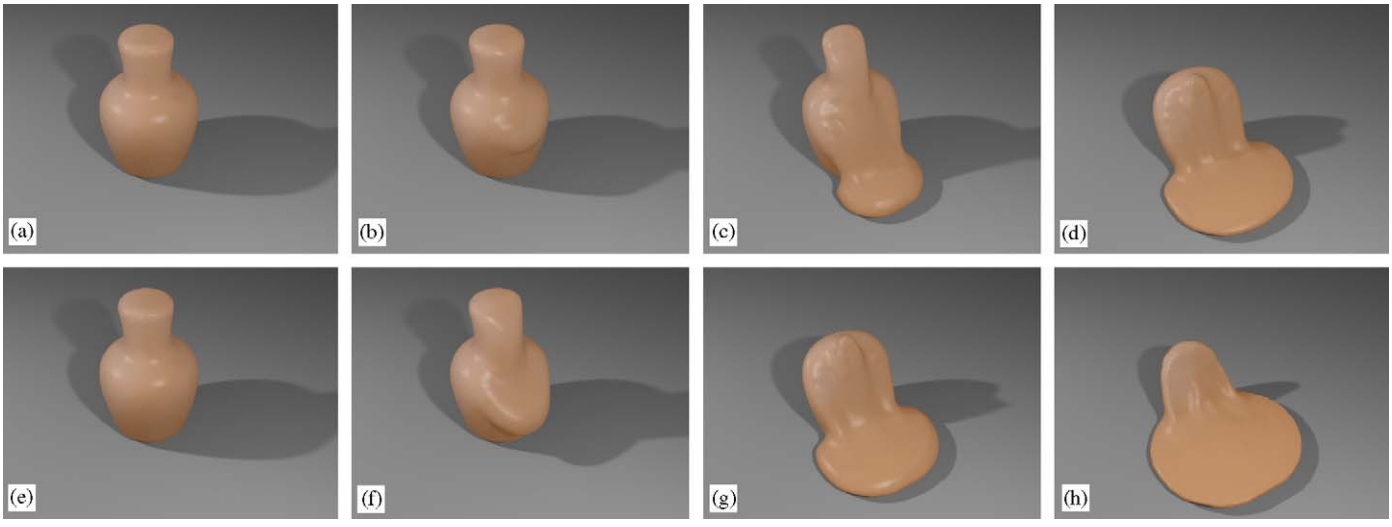


Fig. 4. (a)–(d) Melting of a wax vase without wind effects: (a) original vase; (b)–(d) melting and flowing; (e)–(f) melting of a wax vase with wind: (e) original vase; (f)–(h) melting and flowing. For comparison, each image is generated at the same simulation time as its counterpart in (a)–(d).

grid. Therefore, more accurate and smoother meshes are generated. To achieve realistic rendering results, the geometric meshes of the melting materials, together with the non-meltable internal objects, are rendered by ray tracing using the subsurface scattering method in Maya/MentalRay. For each computational step, an image frame is rendered and movies are generated from these frames to illustrate the resulting dynamics. The movies can be downloaded from our website (http://www.cs.sunysb.edu/~vislab/projects/amorphous/melting).

Fig. 4 illustrates the melting and flowing of a wax vase. Fig. 4(a)–(d) are generated without any air flow. These snapshots represent the procedure: the solid vase (Fig. 4(a)), the beginning of the melting (Fig. 4(b)), more parts being melted (Fig. 4(c)), and the final result (Fig. 4(d)). Heat is transferred in the simulation space with only diffusion. Our method can include air flow effects. Fig. 4(e)–(h) shows a sequence of images of melting with air flow. A wind blows from the heat source and brings heat to the vase. For comparison, each image is generated at the same simulation time as their counterparts in Fig. 4(a)–(d). The melting process is obviously faster due to the wind blowing from the heat source. In Fig. 5, we show the streamlines of the velocity field surrounding the vase in our simulation. With the same heat source position, Fig. 5(a) shows the streamlines created solely by the melted liquid flowing downwards and Fig. 5(b) shows the streamlines generated by both the wind and the liquid at the same time step in Fig. 5(a). The color varies from red to blue as the temperature varies from high to low, respectively. In Fig. 5(b), the vase melts more severely in contrast with Fig. 5(a) due to the blowing wind.

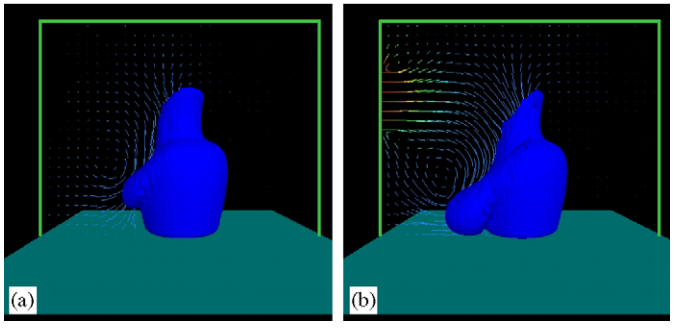


Fig. 5. Velocity field streamlines of a melting vase. (a) No wind; (b) with wind from the heat source.

Fig. 6 shows the melting effects of a volumetric foot obtained from CT scan. The bones of the foot are non-meltable and play a role in the melting as internal objects. The skin and other soft parts of the foot are melting and flowing, while at the same time the bones are revealed. Fig. 7 presents that a volumetric head is melted by a heat source on the right and flows to the bottom. Starting from a CT dataset, the internal mass distribution is heterogeneous. Consequently, the melting behavior illustrates this feature with some observed caves and bulges. The viscosity here is low, which determines the turbulent flowing behavior of the melted liquid. In Fig. 8, we show two Chinese characters attached to a wall, melting and flowing to the ground.

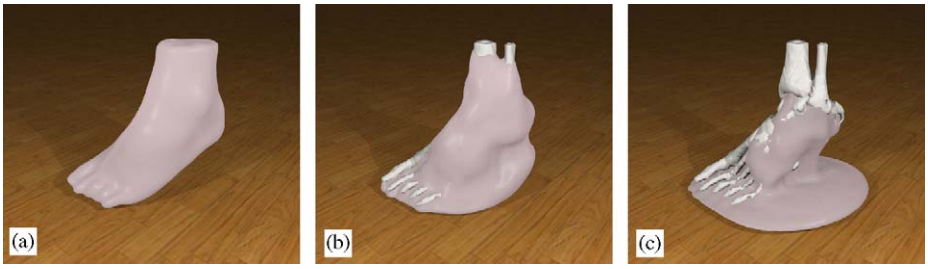


Fig. 6. Snapshots of melting a volumetric foot obtained from CT scan. (a) Original foot; (b) the skin and other soft parts are melted and begin to flow downwards, and the bones are revealed; (c) more bones are revealed.

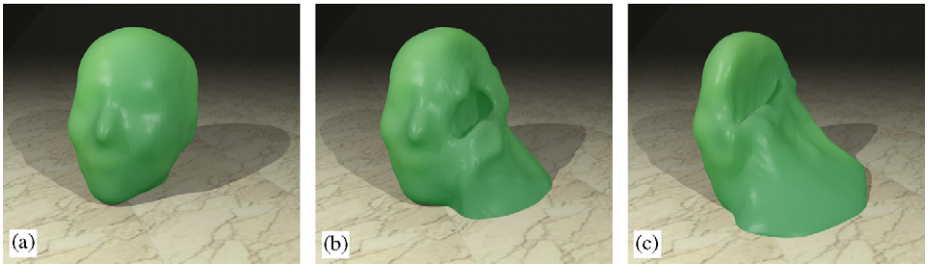


Fig. 7. Snapshots of melting a volumetric head. Starting from a CT dataset, the internal mass distribution is heterogeneous. Consequently, the melting behavior illustrates this feature with some observed caves and bulges. (a) Starting solid head; (b) melting and flowing make the internal cavity visible; (c) half of the head is melted.

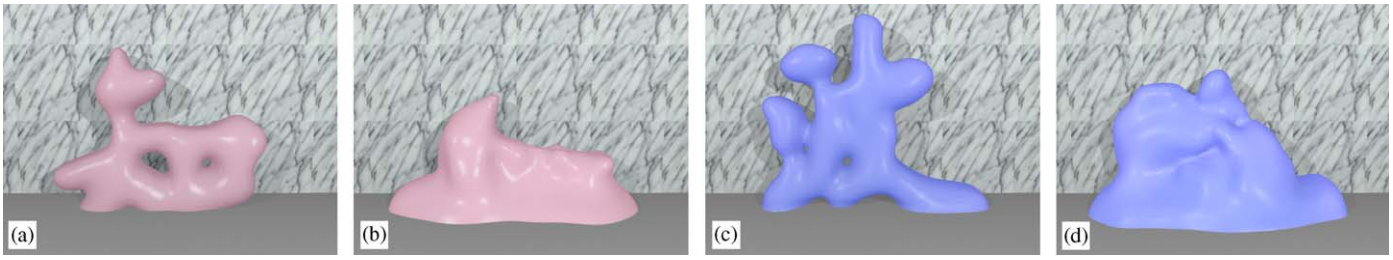


Fig. 8. Snapshots of melting of two Chinese characters. (a) and (c): the two characters are attached to a wall; (b) and (d): due to the heat, the characters are melted and flow down to the floor.

We use the 13-velocity LBM model in 3D to generate these results. The lattice resolution is set as $64^3$. The bounce-back boundary conditions are applied for the six walls. As for the temperature, on the left wall, a constant temperature 0.05 is maintained at a region $(x = 0; y = [41 \ldots 54]; z = [25 \ldots 38])$ to represent a heat source. Other than the heat source, an initial temperature 0.0 is used and the thermal diffusivity $\kappa = 0.1$. The other walls of the computational volume are treated via the adiabatical thermal boundary condition $\partial_{\hat{\mathbf{n}}} T = 0$, where $\hat{\mathbf{n}}$ defines the unit normal outward. The kinetic viscosity $v = 0.6$ for the melting liquid and $v = 0.05$ for the air. All the values we used are in the LBM lattice units, refer to Ref. [26] for details about setting constants and parameters for the thermal LBM. We further refer the reader to our previous work [18] for how to convert the actual values of physical properties (space size, velocity, etc.) to such dimensionless numbers for the LBM configuration.

## 8. Conclusion

We have proposed a lattice-based method to simulate the natural melting and flowing phenomena, involving solid melting, liquid flowing and air circulation as well as the heat exchanges among them. A multiple-relaxation-time lattice Boltzmann method provides a good framework to model the particular multiphase scenario in a uniform computational scheme, including complex internal objects treatment, easy viscosity adjustment, direct body force and heat incorporation. Based on this framework, we have pioneered modifying the streaming–collision rules of the LBM to implement the melting and flowing naturally with implicit interface evolution. The numerical computation of both the LBM and the heat transfer adapts well to graphics hardware with only local and parallel operations. In the future, we will apply this framework to other multiphase phenomena, such as boiling and vaporizing.

## Acknowledgments

This work has been partially supported by NSF Grants CCR-0306438 and ACI-0093157.

## References

[1] Carlson M, Mucha P, Horn R, Turk G. Melting and flowing. Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation; 2002. p. 167–74.
[2] Müller M, Keiser R, Nealen A, Pauly M, Gross M, Alexa M. Point based animation of elastic, plastic and melting objects. Proceedings of the ACM SIGGRAPH symposium on computer animation; 2004. p. 141–51.
[3] Rasmussen N, Enright D, Nguyen D, Marino S, Sumner N, Geiger W. et al. Directable photorealistic liquids. Proceedings of the ACM SIGGRAPH symposium on computer animation; 2004. p. 193–202.
[4] Wei X, Li W, Kaufman A. Melting and flowing of viscous volumes. Proceedings of the computer animation and social agents; 2003. p. 54–9.
[5] Succi S. The lattice Boltzmann equation for fluid dynamics and beyond. Numerical mathematics and scientific computation. Oxford: Oxford University Press; 2001.
[6] Enright D, Marschner S, Fedkiw R. Animation and rendering of complex water surfaces. Proceedings of SIGGRAPH; 2002. p. 736–44.
[7] Foster N, Metaxas D. Realistic animation of liquids. Graphical Models and Image Processing 1996;58(5):471–83.
[8] Foster N, Metaxas D. Modeling the motion of a hot, turbulent gas. Proceedings of SIGGRAPH; 1997. p. 181–8.
[9] Stam J. Stable fluids. Proceedings of SIGGRAPH; 1999. p. 121–8.
[10] Fedkiw R, Stam J, Jensen H. Visual simulation of smoke. Proceedings of SIGGRAPH; 2001. p. 15–22.
[11] Selle A, Rasmussen N, Fedkiw R. A vortex particle method for smoke, water and explosions. Proceedings of SIGGRAPH; 2005. p. 910–4.
[12] Nguyen D, Fedkiw R, Jensen H. Physically based modeling and animation of fire. Proceeding of SIGGRAPH; 2002. p. 721–8.
[13] Mueller M, Charypar D, Gross M. Particle-based fluid simulation for interactive applications. Proceeding of ACM SIGGRAPH/EUROGRAPHICS symposium on computer animation; 2003. p. 154–9.
[14] Premoze S, Tasdizen T, Bigler J, Lefohn A, Whitaker R. Particle-based simulation of fluids. Proceeding of Eurographics; 2003. p. 401–10.
[15] Chen S, Doolen G. Lattice Boltzmann method for fluid flows. Annual Review of Fluid Mechanics 1998;30:329–64.
[16] Chu N, Tai C. Moxi: real-time ink dispersion in absorbent paper. Proceedings of SIGGRAPH; 2005. p. 504–11.
[17] Thürey N, Rüde U. Free surface lattice-Boltzmann fluid simulations with and without level sets. Workshop on vision, modelling, and visualization (VMV Stanford); 2004. p. 199–208.
[18] Wei X, Zhao Y, Fan Z, Li W, Qiu F, Yoakum-Stover S, et al. Lattice-based flow field modeling. IEEE Transactions on Visualization and Computer Graphics 2004;10(6):719–29.
[19] D'Humiéres D, Ginzburg I, Krafczyk M, Lallemand P, Luo L. Multiplerelaxation-time lattice Boltzmann models in three-dimensions. Philosophical Transactions of Royal Society of London 2002;360(1792):437–51.

[20] Qiu F, Zhao Y, Fan Z, Wei X, Lorenz H, Wang J et al. Dispersion simulation and visualization for urban security. Proceedings of IEEE visualization; 2004. p. 553-60.
[21] McCracken M, Abraham J. A multiple relaxation time lattice Boltzmann model for multiphase flow. Physical Review E 2005; 71:036701–10.
[22] Terzopoulos D, Platt J, Fleischer K. Heating and melting deformable models (from goop to glop). Proceedings of graphics interface; 1989. p. 219–26.
[23] Goktekin T, Bargteil A, O'Brien J. A method for animating viscoelastic fluids. Proceedings of SIGGRAPH; 2004. p. 463–8.
[24] Zhu Y, Bridson R. Animating sand as a fluid. Proceedings of SIGGRAPH; 2005. p. 965–72.
[25] Sussman M, Smereka P, Osher S. A level set approach for computing solutions to incompressible two-phase flow. Journal of Computational Physics 1994;114:146–59.
[26] Lallemand P, Luo L. Theory of the lattice Boltzmann method: acoustic and thermal properties in two and three dimensions. Physical Review E 2003;68:036706.
[27] Li W, Wei X, Kaufman A. Implementing lattice boltzmann computation on graphics hardware. The Visual Computer 2003; 19(7–8):444–56.
[28] Mei R, Luo LS, Shyy W. An accurate curved boundary treatment in the lattice Boltzmann method. Journal of Computational Physics 1999;155:307–30.
[29] Zhang R, Chen H. Lattice Boltzmann method for simulations of liquid–vapor thermal flows. Physical Review E 2003;67:066711.
[30] Lorensen W, Cline H. Marching cubes: a high resolution 3d surface construction algorithm. Proceedings of SIGGRAPH; 1987. p. 163–9.
[31] Dupuis A, Chopard B. Theory and applications of an alternative lattice Boltzmann grid refinement algorithm. Physical Review E 2003; 67:066707.