Commun. Comput. Phys.
doi: 10.4208/cicp.2014.m350

Vol. **17**, No. 5, pp. 1201-1224
May 2015

Numerical Simulation of Melting with Natural Convection Based on Lattice Boltzmann Method and Performed with CUDA Enabled GPU

Wei Gong[1,3,*], Kévyn Johannes[2,3] and Frédéric Kuznik[1,3]

[1] *INSA-Lyon,CETHIL, UMR5008, F-69621, Villeurbanne, France.*
[2] *Université Lyon 1, F-69621, France.*
[3] *Université de Lyon, CNRS, France.*

Received 31 October 2013; Accepted (in revised version) 5 September 2014

Abstract. A new solver is developed to numerically simulate the melting phase change with natural convection. This solver was implemented on a single Nvidia GPU based on the CUDA technology in order to simulate the melting phase change in a 2D rectangular enclosure. The Rayleigh number is of the order of magnitude of 10^8 and Prandlt is 50. The hybrid thermal lattice Boltzmann method (HTLBM) is employed to simulate the natural convection in the liquid phase, and the enthalpy formulation is used to simulate the phase change aspect. The model is validated by experimental data and published analytic results. The simulation results manifest a strong convection in the melted phase and a different flow pattern from the reference results with low Rayleigh number. In addition, the computational performance is estimated for single precision arithmetic, and this solver yields 703.31 MLUPS and 61.89 GB/s device to device data throughput on a Nvidia Tesla C2050 GPU.

PACS: 44.35.+c

Key words: LBM, GPU, phase change, natural convection, numerical simulation, experimental study.

1 Introduction

Heat storage technologies now are widely used as an effective means to manage the energy availability and decrease the temperature fluctuations in various practical applications, such as in the electronics industry [1], the automotive industry [2,3] and building design [4,5]. Compared to the sensible heat storage [6], the latent heat storage has a significant advantage as it uses less storage volume to achieve a specified amount of heat

*Corresponding author. *Email addresses:* `wei.gong@insa-lyon.fr` (W. Gong), `Kevyn.Johannes@insa-lyon.fr` (K. Johannes), `frederic.kuznik@insa-lyon.fr` (F. Kuznik) load. Furthermore, in general, latent heat storage process is a constant temperature process.

One of the most important physical phenomenon of latent heat storage is the solid-liquid phase change, which is also a very critical process for many other applications such as in the metal casting industry. To better understand solid-liquid phase change, we need to know the temperature distribution in both solid and liquid phases, the flow characteristics in liquid phase, the heat transfer characteristics and the transient interface location. However, the solid-liquid phase change phenomenon is a complex process which couples the natural convection in the liquid phase, the shifting of the solid-liquid boundary and a heat transfer process. Due to these reasons, the solid-liquid phase change is a strong non-linear process, difficult to analyse except for simple and ideal test cases.

The traditional way to simulate numerically the solid-liquid phase change by solving the Navier-Stokes equations is an effective method [8, 9]. The key aspect being the switching of the advection term on and off during the phase change. In addition, the dynamic position of the solid-liquid interface must be computed at each time step and the corresponding boundary conditions for the LBM must be chosen accordingly.

In the past two decades the lattice Boltzmann method (LBM) matured as a promising approach for CFD due to its intrinsic parallelism and good numerical stability along with favourable numerical dissipation properties [24, 25]. All of those advantages make LBM a promising complementary approach to the direct solution of the Navier-Stokes (NS) equations [26]. In relatively recent studies, some researchers have begun to choose LBM to simulate phase change [10–14].

In order to make the most of the inherent parallelism of LBM, numerous efforts have been devoted to harness the calculation power of graphic processing units (GPU) [15–18]. A GPU is characterised by its highly parallel, manycore structure, which is especially suited for data intensive calculations. Because more transistors are used to perform data operations, a GPU performs considerably more float-point operations per time unit and has a large memory bandwidth than a CPU counterpart [29]. Since GPUs were initially used to render graphics, using them for a general purpose calculation remains complex and prevents an easy implementation.

In this study, an approach is presented to numerically simulate the melting process in combination with natural convection. We developed a highly efficient solver, running on a graphic processing unit, for the liquid-solid melting phase change. The hybrid thermal lattice Boltzmann method is employed to simulate the natural convection in the liquid phase. For phase change, the enthalpy formulation is used. Because the total latent heat of each grid node is a function of temperature and can embody the status of the material, it is used as the key parameter to locate the interface. The simulation program is optimized for parallel computing. Besides, experiments are also carried out, in order to validate the present numerical simulation.

2 Numerical modelling

2.1 Definition of the problem

Phase change materials (PCMs) are used to store or release energy. This process is related to the melting/solidification. In this study, n-octadecane is chosen as PCM of interest; its thermophysical properties used for simulation are presented in Table 1. One important property of the n-octadecane is its high Prandtl number. A rectangular container whose dimensions are $30\,\text{mm}(L)\times 172\,\text{mm}(H)\times 172\,\text{mm}(D)$ filled with n-octadecane — with a height of 152 mm — is heated up from the two vertical sides, the other sides are supposed adiabatic (Fig. 1 and Fig. 7). During melting process, the early stage is governed by conduction. Gradually with liquid fraction increasing, the convection develops and then dominates the heat transfer. In the not-yet-melted solid phase, conduction is still the main heat transfer form. The natural convection in the liquid phase causes the upper part of the PCM to melt faster than the bottom part, and, eventually modifies the shape of the solid-liquid interface. In this work, the Rayleigh number Ra is of the order of 10^8. As a result, the convection plays an important role during the heat transfer. In the next section (Section 2.2), we will present the numerical model construction and then in Section 3 the detailed implementation on GPU.

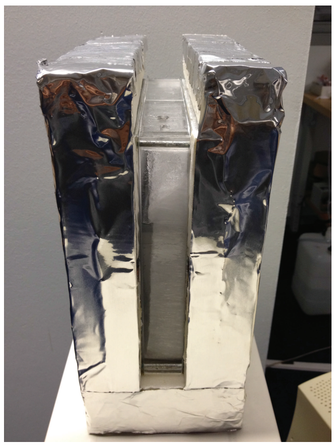


Figure 1: Experiment enclosure.

2.2 Model construction

2.2.1 The governing equations

For the entire melting process, we assume that the liquid is Newtonian and incompressible, and assume that both the solid and the liquid phases have the same heat capacity. According to Voller *et al.* [33], the governing equations that describe melting phase change are in the following forms:

Table 1: Thermophysical properties of n-octadecane used for simulation [7].

Property	**Value**
Melting temperature T_f	$28.0[^\circ\mathrm{C}]$
Latent heat λ	$1.25\times 10^5[\mathrm{J/kg}]$
Specific heat C_p	$1250[\mathrm{J/(kg\cdot ^\circ C)}]$
Thermal conductivity k	$0.2[\mathrm{W/(m\cdot K)}]$
Density ρ	$800[\mathrm{kg/m^3}]$
Kinematic viscosity ν	$1\times 10^{-5}[\mathrm{m^2/s}]$
Thermal expansion coefficient β	$0.002[\mathrm{K^{-1}}]$
Thermal diffusivity α	$2\times 10^{-7}[\mathrm{m^2/s}]$

Continuity equation:

$$\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho\vec{u})=0; \tag{2.1}$$

Momentum conservation:

$$\frac{\partial(\rho\vec{u})}{\partial t}+\nabla\cdot(\rho\vec{u}\vec{u})=-\nabla p+\nabla\cdot(\mu\nabla\vec{u})+\vec{S_1}+\vec{S_2}; \tag{2.2}$$

Energy conservation:

$$\frac{\partial\left(\rho C_p T\right)}{\partial t}+\nabla\cdot\left(\rho C_p T\vec{u}\right)=\nabla\cdot(k\nabla T)+S_3, \tag{2.3}$$

where μ is the dynamic viscosity, C_p, k, $\vec{u}$ are specific heat, heat conductivity and macroscopic velocity. $\vec{S_1}$, $\vec{S_2}$ and S_3 are source terms due to the presence of phase change coupled by natural convection.

In the governing equations, because of the aforementioned existence of phase change, the critical point to be determined during the simulation is whether the advection terms will appear or not. In order to achieve this requirement, Brent *et al.* [19] devises the source terms $\vec{S_1}$ in momentum equation to have a form such as: $\vec{S_1}=A\vec{u}$. Moreover, the second source term S_2 is introduced by buoyancy, which is $S_2=\rho g\beta(T-T_r)$ and T_r is a reference temperature. The momentum equation is then rewritten as:

$$\rho\frac{\partial\vec{u}}{\partial t}+\rho(\vec{u}\cdot\nabla)\vec{u}=-\nabla\cdot p+\mu\nabla^2\vec{u}+A\vec{u}+\rho g\beta(T-T_r). \tag{2.4}$$

A can be described by two forms: Carman-Kozeny form or simplified linear form (Eq. (2.5)). The symbol l is the liquid fraction, ϵ' and C are two simulation dependent constants, the former is used to avoid the division by zero and the latter to describe the morphology of the porous solid-liquid interface [19]. By using this definition of A, the term $A\vec{u}$ varies gradually between 0 and a relatively high value according to the liquid fraction in a cell. As a consequence (depending on the magnitude of constant C), $A\vec{u}$

dominates the other terms in the momentum equation. From a physical point of view, this process depicts the velocity evolution during the phase change

$$A=\begin{cases} -C\dfrac{(1-l)^2}{(l^3+\epsilon')} & \text{Carman-Kozeny form,} \\ -C(1-l) & \text{simplified linear form.} \end{cases} \tag{2.5}$$

The source term S3 in the energy equation will be analyzed in Section 2.2.3. In the current simulation, the continuity equation and momentum equation will be solved by lattice Boltzmann method and the energy equation by finite differences.

2.2.2 Lattice Boltzmann equation

The lattice Boltzmann equation is initially derived from the lattice gas automata (LGA) theory, and it is also considered as a specially discretized form of the Boltzmann equation in time, space and particular velocity space [35]. The LBM is a mesoscopic CFD method which keeps some simplified microscopic molecular dynamics properties to recover the macroscopic hydrodynamics results. In the LBM, the physical region is divided into regular spacial lattices, each of which is connected to its neighbours through a set of predefined molecule velocities' vectors $\vec{e}_\alpha$, where $\alpha(=0,1\cdots,n)$ is the quantity of discrete velocities, and for the still molecules, $\vec{e}_0=0$. It describes the temporal evolution of the molecule distribution functions f_α as a result of the collision and the propagation (Eq. (2.6))

$$\underbrace{|f_\alpha(\vec{x}+\vec{e}_\alpha\delta t,t+\delta t)\rangle-|f_\alpha(\vec{x},t)\rangle}_{Propagation}=\underbrace{\Phi}_{Collision}, \tag{2.6}$$

in which, $\vec{x}$ is the position and δt is the time step.

The broadly employed single relaxation time LBGK model is:

$$|f_\alpha(\vec{x}+\vec{e}_\alpha\delta t,t+\delta t)\rangle-|f_\alpha(\vec{x},t)\rangle=-\frac{1}{\tau'}\left[|f_\alpha(\vec{x},t)\rangle-|f_\alpha^{eq}(\vec{x},t)\rangle\right], \tag{2.7}$$

where τ' is the relaxation time.

The equilibrium distributions f_α^{eq} are calculated by:

$$f_\alpha^{eq}=\rho\omega_\alpha\left[1+\frac{e_\alpha\cdot\vec{u}}{c_s^2}+\frac{(e_\alpha\cdot\vec{u})^2}{2c_s^4}-\frac{u^2}{2c_s^2}\right], \tag{2.8}$$

where ω_α is the weight coefficient, $c_s^2=RT$ is the sound speed, and $\vec{u}$ is the macroscopic velocity. In addition, the macroscopic density and momentum are given by:

$$\rho=\sum_\alpha f_\alpha \quad\text{and}\quad \rho\vec{u}=\sum_\alpha \vec{e}_\alpha f_\alpha. \tag{2.9}$$

In the LBGK model, each physical value evolves towards its equilibrium state with the same relaxation time τ', which means the LBGK model has a unchangeable unit Prandtl

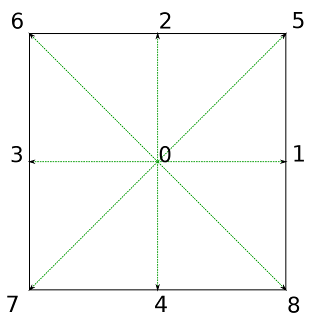


Figure 2: D2Q9 scheme.

number and thus the transports of momentum and heat are non-adjustable. This restriction makes the LBGK model less flexible as for a CFD method. To address this problem, Chen et al. proposed a method to make the Prandtl number adjustable within a range (from 0.59 to 3.3) when the small temperature difference limit satisfied [37].

D'Humières proposed an another collision model with multiple-relaxation-time (MRT). In the MRT model, the discrete velocities (e_α) are identical as in the LBGK model, whereas the different moments — relating to macroscopic physical values — have different relaxation times. The MRT model is mathematically more complex than LBGK, however it is proved to possess better numerical stability [21].

In the current simulation, we choose the D2Q9 discrete velocity mode (two dimensions, nine predefined propagation directions, see Fig. 2). The $\vec{e}_\alpha$ are given as:

$$\vec{e}_\alpha = \begin{cases} (0,0) & \alpha = 0, \\ e(\cos[(\alpha-1)\pi/2], \sin[(\alpha-1)\pi/2]) & \alpha = 1-4, \\ e(\cos[(2\alpha-9)\pi/4], \sin[(2\alpha-9)\pi/4]) & \alpha = 5-8. \end{cases} \tag{2.10}$$

The updating rule of the MRT-LBM is given by Eq. (2.11) [21], the vector $|f_\alpha\rangle$ is mapped to a moment vector space by a well-designed linear transformation (Eq. (2.12) and Eq. (2.13)). Within the D2Q9 stencil, the moment vector $|m_\alpha\rangle$ is described by Eq. (2.12) [24], where ρ is the density, e is related to energy, ϵ is a fourth order moment related to energy square, j_x, j_y are momentums in x and y directions respectively, q_x, q_y are related to the energy flux in x, y directions, and p_{xx}, p_{xy} are higher order terms related to the stress tensor. In the MRT model, the collision process actually takes place in moment space; after collision, the moments are mapped back to distribution functions; and then the propagation is performed within particle density space.

$$|f_\alpha(\vec{x}+\vec{e}_\alpha\delta t, t+\delta t)\rangle - |f_\alpha(\vec{x},t)\rangle = -\boldsymbol{M}^{-1}\cdot\boldsymbol{S}\cdot\left[|m_\alpha(\vec{x},t)\rangle - |m_\alpha^{eq}(\vec{x},t)\rangle\right], \tag{2.11}$$

$$|m_\alpha\rangle = \boldsymbol{M}|f_\alpha\rangle = \left(\rho, e, \epsilon, j_x, q_x, j_y, q_y, p_{xx}, p_{xy}\right)^\top, \tag{2.12}$$

$$\boldsymbol{M}=\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -4 & -1 & -1 & -1 & -1 & 2 & 2 & 2 & 2 \\ 4 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & 0 & 1 & -1 & -1 & 1 \\ 0 & -2 & 0 & 2 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 & 1 & 1 & -1 & -1 \\ 0 & 0 & -2 & 0 & 2 & 1 & 1 & -1 & -1 \\ 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \end{pmatrix}. \tag{2.13}$$

In Eq. (2.11), $|m_\alpha^{eq}\rangle$ are the equilibrium quantities. For the conserved quantities, $m_0^{eq}=\rho^{eq}=\rho$, $m_3^{eq}=j_x^{eq}=j_x$ and $m_5^{eq}=j_y^{eq}=j_y$; and for the other non-conserved values, equilibrium states are calculated by Eq. (2.14) [21]:

$$m_1^{eq}=e^{eq}=-2\rho+3\left(j_x^2+j_y^2\right),\quad m_2^{eq}=\epsilon^{(eq)}=\rho-3\left(j_x^2+j_y^2\right), \tag{2.14a}$$

$$m_4^{eq}=q_x^{(eq)}=-j_x,\quad m_6^{eq}=q_y^{(eq)}=-j_y, \tag{2.14b}$$

$$m_7^{eq}=p_{xx}^{(eq)}=\frac{1}{3}\left(j_x^2-j_y^2\right),\quad m_8^{eq}=p_{xy}^{(eq)}=\frac{1}{3}j_xj_y. \tag{2.14c}$$

The sound speed in lattice units is $c_s=1/\sqrt{3}$.

In Eq. (2.11), **S** is the collision matrix, which is a diagonal square matrix:

$$\begin{aligned}\boldsymbol{S}&=\mathrm{diag}\left(s_\rho,s_e,s_\epsilon,s_{j_x},s_{q_x},s_{j_y},s_{q_y},s_{p_{xx}},s_{p_{xy}}\right)\\ &=\mathrm{diag}(0,s_2,s_3,0,s_5,0,s_7,s_8,s_9),\end{aligned} \tag{2.15}$$

each diagonal element of $\vec{S}$ is one relaxation factor for different moments. According to the reference [24], s_8 and s_2 are controlled by the shear viscosity ν

$$s_8=\frac{2}{6\nu+1}, \tag{2.16}$$

and by the bulk viscosity ζ

$$s_2=\frac{2}{12\zeta+1}. \tag{2.17}$$

Furthermore,

$$s_5=s_7=\frac{8(2-s_8)}{(8-s_8)}\quad\text{and}\quad s_9=s_8. \tag{2.18}$$

In addition, as long as they are within the interval (0,2) [24], s_2 and s_3 can be chosen arbitrarily to enhance the stability of the simulation.

Finally, the collision matrix $\vec{S}$ we used for the simulation is

$$\vec{S}=\mathrm{diag}\left(0,\ 1.64,\ 1.54,\ 0,\ \frac{8(2-s_8)}{(8-s_8)},\ 0,\ \frac{8(2-s_8)}{(8-s_8)},\ s_8,\ s_8\right). \tag{2.19}$$

The LBE can be numerically very stable when solving isothermal flow. However, as solving thermal-hydrodynamics problems, the thermal LBE (TLBE) models are more vulnerable to numerical instabilities. These instabilities are attributed to a spurious algebraic coupling between the shear and energy modes of the linearised evolution operator. This spurious algebraic coupling is rooted in the basic features of LBE. As a result, in order to improve the numerical robustness, Lallemand and Luo proposed a hybrid thermal lattice Boltzmann equation (HTLBE) and proved that it has better numerical stability than the other energy-conserving Boltzmann models [36]. In the HTLBE model, the mass and momentum conservations are solved by MRT-LBE and the temperature by finite differences [22, 23]. Through the Chapman-Enskog analysis, we can successfully recover the macroscopic conservation equations (Eq. (2.1), Eq. (2.2)) from this HTLBE. In the current simulation, as a result, the HTLBE model is employed.

2.2.3 Energy equation

Phase change

During melting, the heat is stored in the form of latent heat of fusion. Therefore, the total energy h is the sum of sensible heat and latent heat as given in Eq. (2.20). We assume that the heat capacity of liquid and solid phases are equal, denoted by C_p. L refers to the latent heat content.

$$h=C_pT+L, \tag{2.20}$$

with $L=l\cdot\lambda$, and l is the liquid fraction.

The energy equation in the form of total energy (h) is given by:

$$\frac{\partial \rho h}{\partial t}+\nabla\cdot(\rho h\vec{u})=\nabla\cdot(k\nabla T). \tag{2.21}$$

The thermal conductivity k is treated as constant during the melting. As Boussinesq approximation is assumed, the density temperature dependence is only taken into account in the buoyancy force term.

After mathematical operations on Eq. (2.20) and Eq. (2.21), the energy equation becomes:

$$\frac{\partial T}{\partial t}+\nabla\cdot(\vec{u}T)=\alpha\nabla^2T-S_3, \tag{2.22}$$

where S_3 is:

$$S_3=\frac{\partial(L/C_p)}{\partial t}+\nabla\cdot\Big(\frac{\vec{u}L}{C_p}\Big). \tag{2.23}$$

S_3 comprises the effect of latent heat change. As stated before, the material during melting phase change absorbs the heat and stores it as latent heat of fusion in a narrow range of temperature. This can be considered as a heat sink as described by Eq. (2.23).

In the solid and liquid phase, the term $\nabla\cdot(\frac{\vec{u}L}{C_p})$ is equal to 0, while along the liquid-solid interface, this term is not 0 but sufficiently small to be neglected.

In order to keep the units consistent with LBM, the energy equation must be transformed to dimensionless form, by choosing the following reference quantities and dimensionless numbers:

$$u_{ref} = \alpha\sqrt{Ra}/H, \quad T_{ref} = \Delta T = T_h - T_f, \quad H_{ref} = H,$$
$$Ra = \frac{g\beta\Delta T H^3}{\nu\alpha}, \quad Pr = \frac{\nu}{\alpha}, \quad Ste = \frac{C_p\cdot(T_h - T_c)}{\lambda}.$$

Then the dimensionless form of the energy equation is

$$\frac{\partial T^*}{\partial t^*} + \nabla^*\cdot(\vec{u}^* T^*) = Ra^{-\frac{1}{2}}\nabla^{*2}T^* - S_3^*, \tag{2.24}$$

with the source term S_3^*:

$$S_3^* = \frac{1}{\rho\cdot Ste}\frac{\partial l}{\partial t^*}, \tag{2.25}$$

where t^* is the dimensionless time, and the liquid fraction l is defined as

$$l^N = \begin{cases} 1, & T^{*N} \geq T_f^* + \frac{1-l^{N-1}}{Ste}, \\ 0, & T^{*N} \leq T_f^* - \frac{l^{N-1}}{Ste}, \\ (T^{*N} - T_f^*)\cdot Ste + l^{N-1}, & \text{otherwise}. \end{cases} \tag{2.26}$$

Consequently, the temperature at time step n is used to update liquid fraction l, in order to get the source term S_3^* in Eq. (2.24), then using value S_3^* to calculate the temperature in time step $n+1$ and so on.

Discretization

For coupling with LBM, the energy equation furthermore needs to be discretized. During this step, the reference length and time are selected as the lattice size δx and time step δt. Because the lattices are in the form of a square, so $\delta y = \delta x$. Based on those two reference values, the computation domain has a height in lattice unit $N_y = 1/\delta x$ and the maximum time steps to achieve the prescribed melting process is $t^*/\delta t$.

The energy equation in lattice units is:

$$\frac{\partial T_{lb}}{\partial t_{lb}} + \nabla\cdot(\vec{u}_{lb}T_{lb}) = \frac{\delta t}{\delta^2 x}Ra^{-\frac{1}{2}}\nabla^2\cdot T_{lb} - \frac{1}{\rho\cdot Ste}\frac{\partial l}{\partial t_{lb}}, \tag{2.27}$$

where the subscript lb means the lattice units, and $T_{lb} = T^*$, $\rho = 1$.

The finite difference method is used to discretize the diffusion and advection terms. As suggested by Lallemand and Luo [24], for a D2Q9 simulation, the nine point finite-difference operators are used to get the gradients and Laplacian:

$$\begin{aligned}\partial_x T_{lb}\,|_{(i,j)} = & T_{lb}(i+1,j) - T_{lb}(i-1,j) - \frac{1}{4}[T_{lb}(i+1,j+1) \\ & - T_{lb}(i-1,j+1) + T_{lb}(i+1,j-1) - T_{lb}(i-1,j-1)],\end{aligned} \tag{2.28}$$

$$\begin{aligned}\partial_y T_{lb}\,|_{(i,j)} = & T_{lb}(i,j+1) - T_{lb}(i,j-1) - \frac{1}{4}[T_{lb}(i+1,j+1) \\ & - T_{lb}(i+1,j-1) + T_{lb}(i-1,j+1) - T_{lb}(i-1,j-1)],\end{aligned} \tag{2.29}$$

$$\begin{aligned}\nabla^2 T_{lb}\,|_{(i,j)} = & 2[T_{lb}(i+1,j) + T_{lb}(i-1,j) + T_{lb}(i,j+1) + T_{lb}(i,j-1)] - \frac{1}{2}[T_{lb}(i+1,j+1) \\ & + T_{lb}(i-1,j+1) + T_{lb}(i-1,j-1) + T_{lb}(i+1,j-1)] - 6T_{lb}(i,j).\end{aligned} \tag{2.30}$$

2.2.4 Problem solving

The melting goes through three stages: (1) pure solid phase; (2) melting stage; (3) pure liquid phase. In the pure liquid stage, there exists the natural convection. The temperature equation and the LBE must be coupled by adding buoyancy $F = \sqrt{g\beta\Delta T_{lb}}$ to the momentum along y direction (j_y). In solid phase stage, there is no flow, so the calculation of LBE must stop. As a result, the simulation has to do three steps successively during each time step: checking the liquid fraction of each lattice; calculating the temperature; deciding whether calculating the velocity field or not according to the liquid fraction of each lattice.

3 Nvidia GPU hardware and implementation

This section addresses two topics. The first part gives a concise introduction to the Nvidia CUDA technology. The second one presents the implementation of the algorithm.

3.1 GPU with CUDA architecture

3.1.1 CUDA program pattern

A modern GPU is characterized by its parallel computing ability, its high data throughput as well as its programmable features. However it was difficult to perform general purpose computations on GPUs before the release of the Nvidia's CUDA technology. The CUDA is a general purpose parallel computing technology, which comes with a set of development tools such as high level language compilers. In this paper, The CUDA C is employed.

The code written using CUDA C is of two types:

- Host code, which consists of functions running on the CPU;
- Device code, which includes: (1) functions (known as *kernel*) invoked by host functions and (2) device functions invoked by kernels and other device functions. Device code is executed on the GPU.

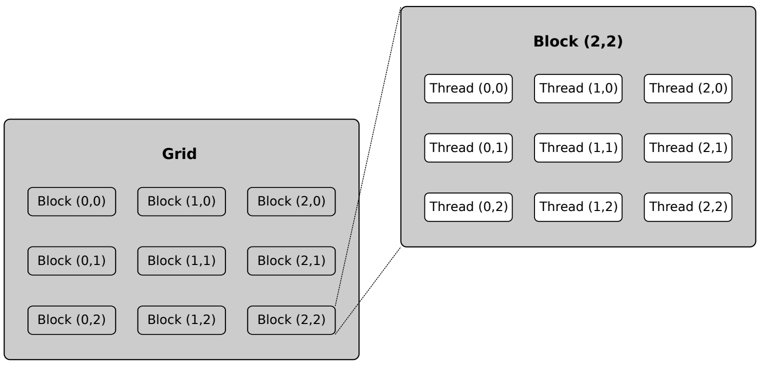


Figure 3: CUDA Block of threads and grid of blocks.

Kernels are launched and executed in parallel by designating an execution pattern through a set of *threads*. The threads are bundled in the form of *thread blocks*. Blocks with equal size are organized into *grids*, see Fig. 3. Threads and blocks are indexed with two built-in up to 3-dimensions variables: `threadIdx` and `blockIdx`.

3.1.2 Memory types

Depending on the accessibility to the memory by kernels, the total memory space on the GPU is categorized by registers, shared memory, constant memory, texture memory and global memory, see Fig. 4. Global memory can be accessed by each thread during the

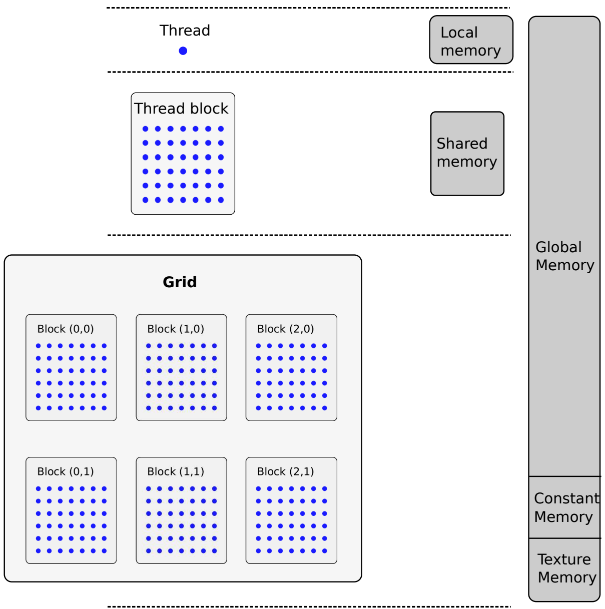


Figure 4: Memory structure of GPU.

runtime of the kernel. Global memory is quantitatively abundant but has the slowest access speed. Therefore, avoiding unnecessary global memory readings and writings should be kept in mind when programming. Registers are used to store local variables. They are not addressable and programmers need to pay attention to prevent them from spilling. Shared memory is used to store the local variables which are declared as shared. It is addressable and as fast as register.

In addition, there are two types of read-only memory that can be accessed by all the threads during the execution of a kernel, one of which being constant memory. The constant memory space is included and addressed in device memory. However, this memory is cached.

3.1.3 CUDA hardware

A CUDA enabled GPU contains several Streaming Multiprocessors (SMs). Each SM has its own register, shared memory, constant cache, as well as several CUDA cores. The SM processes the threads within a block by groups of 32, called *warps*. In this study, we use a Tesla C2050 Nvidia GPU. Its technical specifications are shown in Table 2.

Table 2: Specifications of the Tesla C2050.

CUDA Capability	2.0
Number of SMs	14
Number of CUDA core per SM	32
Constant memory	65536 bytes
Shared memory per block	49152 bytes
Registers available per block	32768
Global memory	3072 MB
Maximum number of threads per SM	1536
Maximum number of threads per block	1024
Maximum dimension of a block	$1024\times1024\times64$
Maximum dimension of a grid	$65535\times65535\times65535$

3.2 Implementation

Fig. 5 shows the diagram of nodes. The blue frame embodies the physical boundary. The nodes enclosed by the blue frame are the calculation domain.

The lattices' topology in the computation domain must be mapped to the threads of the execution grid on the GPU [28]. This fulfills the requirement of thread-level parallelism of CUDA GPU [29]. For our 2-dimensional simulation ($LX\times LY$), a one dimensional block of size LX and one dimensional grid of size LY are chosen. We set LX being the major dimension and being a multiple of the warp size, which can provide us a suf-

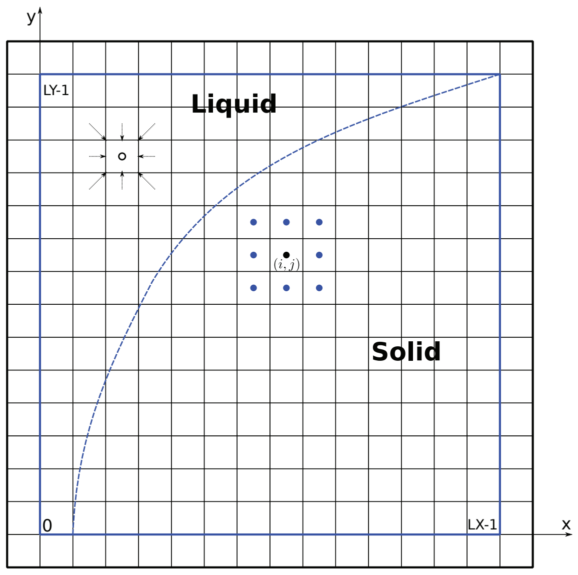


Figure 5: The propagation and temperature scheme.

ficient threads occupancy rate, a coalescence of global memory access and an easy index to the nodes [30].

3.2.1 Data and memory utilization

The global memory is accessed via 32, 64, or 128 byte aligned memory transactions. The warp coalesces the treads memory accesses into one or more memory transactions according to the word size accessed by a single thread and the distribution of the memory addresses. In addition, for the GPU with capability 2.0, the global memory is cached in L1 and L2 by default[†]. The memory is accessed via 128 byte transactions if cached in both L1 and L2; and is accessed via 32 byte transactions if just cached in L2. As a result, a scattered memory addresses distribution across the threads will result in many unused data transactions and less cache hits, which can significantly impact the memory throughput. In this simulation, for each node, nine velocities, one liquid fraction and one temperature value need to be stored in the global memory, which represents eleven single-precision floating-point variables per node. We use a single memory block partitioned into three parts in order to store the distributions array, the temperature array and the liquid fraction array (see Fig. 6).

The energy equation is solved by finite differences. Because the data stored in the shared memory automatically broad themselves to all the threads in a block, in order to decrease the global memory transactions, we use the shared memory to store the temperatures.

[†]For GPU with capability 3.x, the global memory accesses are cached on L2 and for capability 3.5 and higher, can also be cached in the read-only data cache

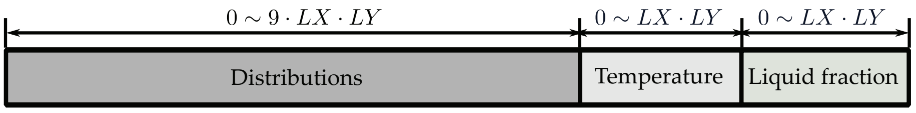


Figure 6: Data structure.

In the LBM implementation, the propagation step also requires the neighbor's relevant distributions information, but all those information are local to each thread, thus just need to be stored in registers. As described in Ref. [30], we also utilize in-place propagation scheme to perform propagation in this simulation (see circle point in Fig. 5).

For the phase change, the liquid fraction information is also stored in registers. In the liquid phase, the LBM collision and propagation are fully performed, and in energy equation, the advection term is valid. In the solid phase, the LBM collision and propagation are disabled, and distributions swap with their opposite counterparts. The macro velocities are set to zero and thus the advection term in energy equation disappears.

3.2.2 Boundary conditions and simulation procedure

On the physical walls (blue frame in Fig. 5), for the LBM, we choose the halfway bounce back scheme. The temperature boundary conditions are applied on the physical walls, which are between two lattices. The boundary condition of the bottom side is:

$$T_{i,-1} = \frac{21T_{i,0} + 3T_{i,1} - T_{i,2}}{23}, \tag{3.1}$$

and of the left vertical wall is:

$$T_{-1,j} = \frac{8T_h - 6T_{0,j} + T_{1,j}}{3}. \tag{3.2}$$

Besides, the $T_{i,LY}$ and $T_{LX,j}$ can be calculated in the same way.

The melting process simulation obeys the following procedure:

1. Initialize model;
2. Execute the propagation, and transfer the current lattice's temperature and its neighbors' temperatures to shared memory;
3. Apply boundary conditions;
4. Compute the liquid fraction and define the node status: liquid or solid;
5. **if the node status is liquid:**
    - Compute the moments of the MRT model, add half of the buoyancy force $(0.5\sqrt{g\beta\Delta T_{lb}})$ to momentum component on y direction j_y;
    - Collide;

- Add the second half of the buoyancy force to the momentum component j_y;
- Compute temperature using the discretized energy equation with advection term;

else:

- Swap distributions f_α on their proper opposite directions;
- Compute the temperature using the discretized energy equation without advection term;

6. Map moment space to discrete velocity space;

7. Update each node for the next iteration.

4 Results and discussion

This section addresses the model validation (Section 4.1), the results of heat transfer and natural convection during melting (Section 4.2) as well as the performance of this parallel implementation (Section 4.3).

4.1 Model validation

4.1.1 Experiment configuration

Experimental data are used for the model validation. The experiment was performed on a transparent enclosure filled with *n*-octadecane (see Fig. 1 and Fig. 7(a)). The diagram of the experiment equipment is shown in Fig. 7(b). A CCD camera and a laser beam were connected by a synchronizer to capture images of the melting process. No thermocouple was embedded in the cavity in order to avoid impacting the natural convection flow in the liquid phase. Instead, we chose to use two heat flux meters on the heating sides of this enclosure, thus guaranteeing the reliability of average Nu number evolution along the walls. The other sides apart from the top and the front were well isolated using asbestos and thereby considered as adiabatic. Both captured images and heat flux data were registered by a computer controlled by the LaVision and the Labview respectively. Experimental parameters are listed in Table 3.

Table 3: Experimental parameters.

T_h [°C]	T_c[°C]	H[m]	L[m]
35.0	27.8	0.152	0.03

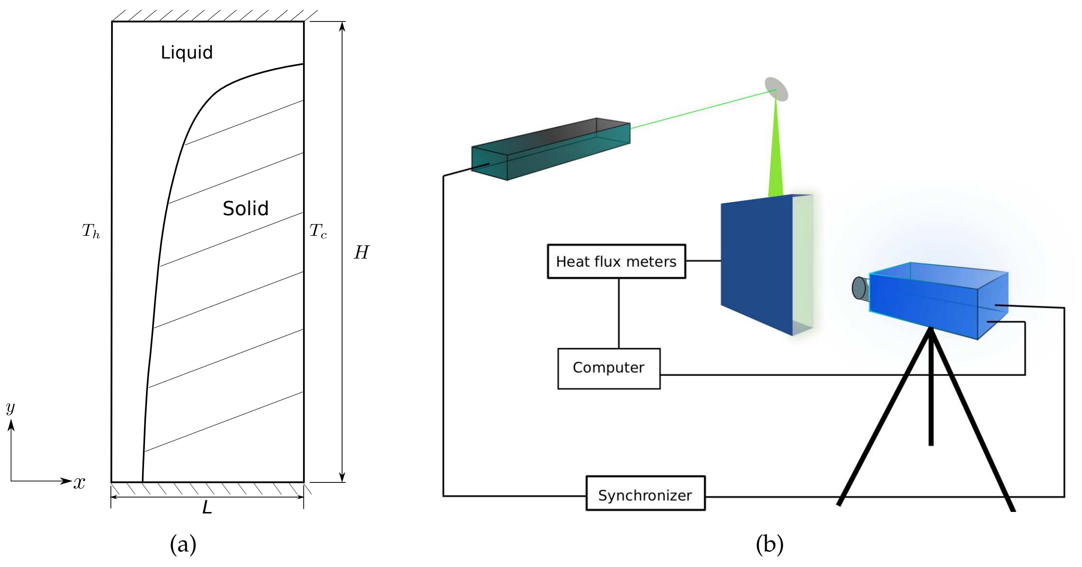


Figure 7: Experiment configuration and diagram.

4.1.2 Model validation

The convergence test was performed under different mesh resolutions. Fig. 8 displays the relative differences of Nu number between the simulation and the analysis [20] at time $\tau = 0.00312$ (5000 s). All the simulations were carried out with a fixed viscosity of 0.01 and the Mach number equal to 0.1. A satisfactory quadratic convergence in space is achieved by the model.

In Fig. 9, melting interface positions are shown at five different instants. Both experimental data and simulation results are displayed. The conversion between the physical time and the dimensionless time ($\tau = FoSte$) is expressed by $t = \left(H^2 \cdot FoSte/\alpha\right)$. The simulation curves are in good agreement with the experimental points. At the onset of melting, n-octadecane is solid, so the melting is just driven by pure conduction. The first curve at dimensionless time equal to $0.00022(353.0\text{s})$ is parallel to the vertical heating wall. On the bottom part of the interface, when $0.1 < y < 0.6$, the simulation results embody the position of the interface. Notwithstanding, the upper portion of the curves retreats slightly more slowly than in the experimental counterparts. This might be due to the fact that the top of the cavity is not perfectly insulated.

Fig. 10 shows the average Nusselt number of the hot vertical wall. The Nusselt number is defined as the amount of heat transferred:

$$Nu = -\int_0^1 \left(\frac{\partial T^*}{\partial x^*}\right) \mathrm{d}y^*.$$

The Rayleigh number is $Ra = 2.48 \times 10^8$ and the Prandtl number is $Pr = 50$. The black long-dashed line is from the correlation by Bejan et al. [20]. All the three sets of results

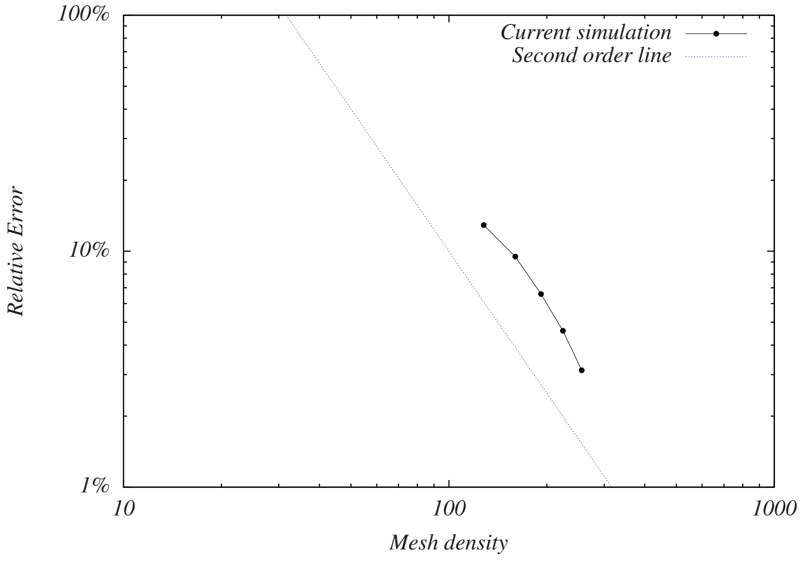


Figure 8: Grid dependence.

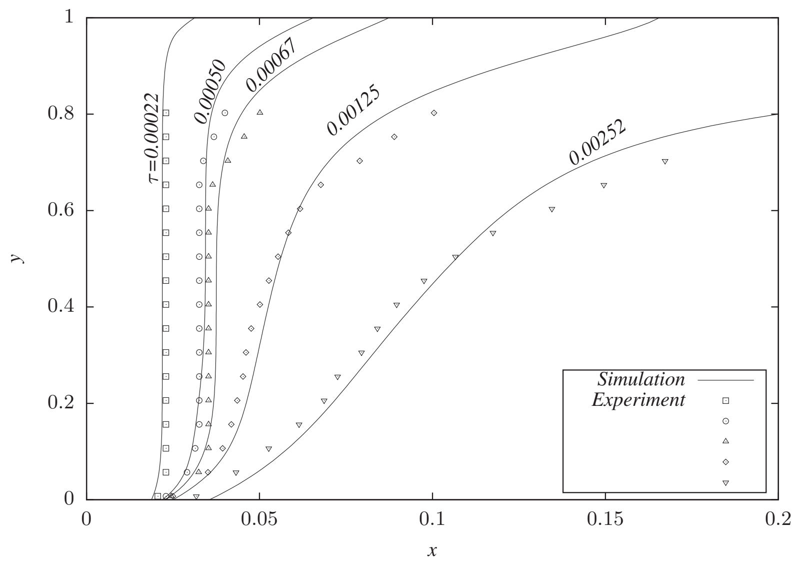


Figure 9: Evolution of melting interface with the dimensionless time: comparison between simulation and experiment. $Ra=2.48\times10^8$, $Pr=50$, $Ste=0.072$.

follow the same pattern. In the beginning, the average Nu starts with a high value, and decreases rapidly with the same order of $\tau^{-1/2}$ as the pure conduction, known as simplified solution of Stefan problem. Later, when the natural convection emerges, the decrease of the Nusselt number becomes moderate, as shown by the curves' gradients on Fig. 10.

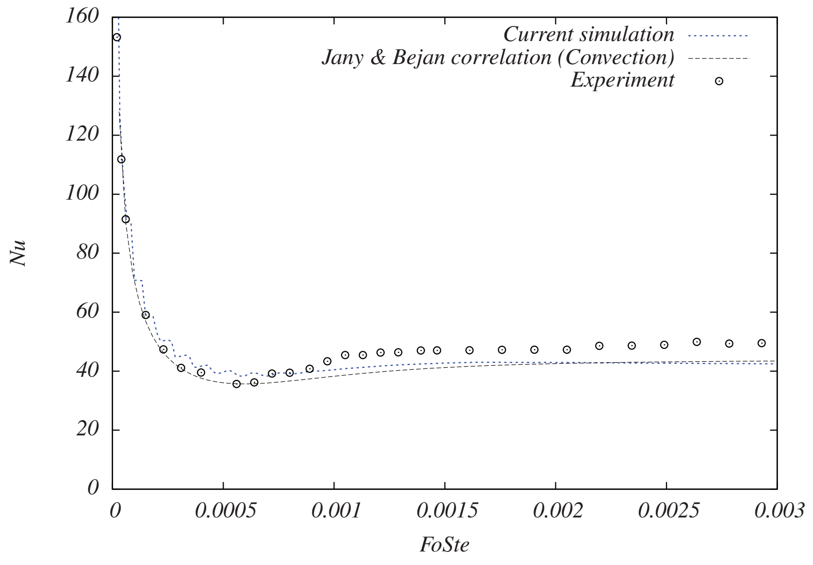


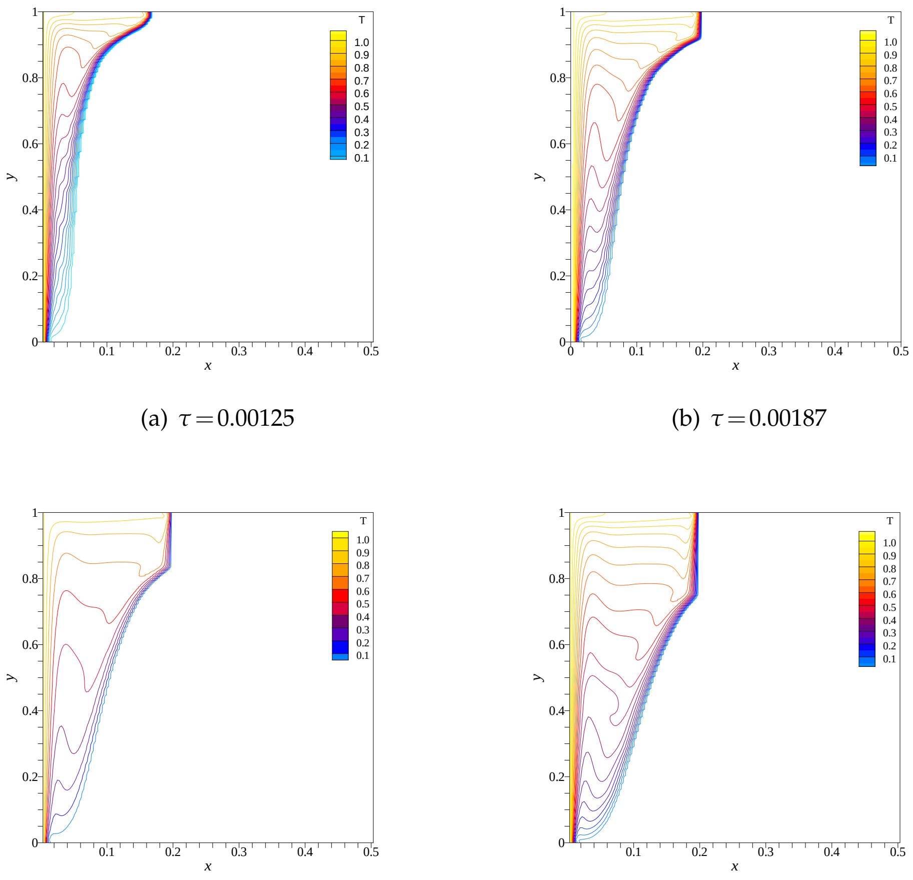
Figure 10: Average Nu number on the left vertical isothermal wall (T_h) during melting of n-octadecane. $Ra=2.48\times10^8$, $Pr=50$, $Ste=0.072$.

Within this elapse of time, the conduction and convection not only exist simultaneously but also exert the same influence on the heat transfer effect. As aforementioned, when convection dominates the melting process, Nu becomes stationary with respect to time. Comparing with Fig. 9, when $\tau=0.00022$, the solid-liquid interface has already exhibited the existence of convection, at the same time, Nu begins its transition.

4.2 Simulation results

Fig. 11 shows the temperature contours at different instants. In most published papers, simulations of melting process driven by natural convection are limited, within low Ra with order of magnitude of 10^7 and low Pr number [31–33]. From the contours, the convection drags the heated melted n-octadecane to the top and the liquid phase is stratified. However, this stratification is different from the results of Okada [27], in which the contour of temperature in the middle of the liquid cavity is parallel to the horizontal wall and the Ra is of order 10^6. In contrast here in Fig. 11(a), the contours are more prone to be vertical. The reason is the higher Ra number in our study and thus a larger degree of convection: as soon as the melted n-octadecane is heated up, the buoyancy force drives it upwards. At the same time, since at the bottom part the flow cavity is quite narrow, the flow rises fast.

From Fig. 12(a), we notice there are two eddies appearing at the height about 0.65 and 0.80. The up rising liquid ultimately reaches the top wall, and turns back. However, when it encounters the newly melted uprising liquid, there is not enough space to let it go down. With the melting developing, the liquid cavity enlarges, and one eddy disap-



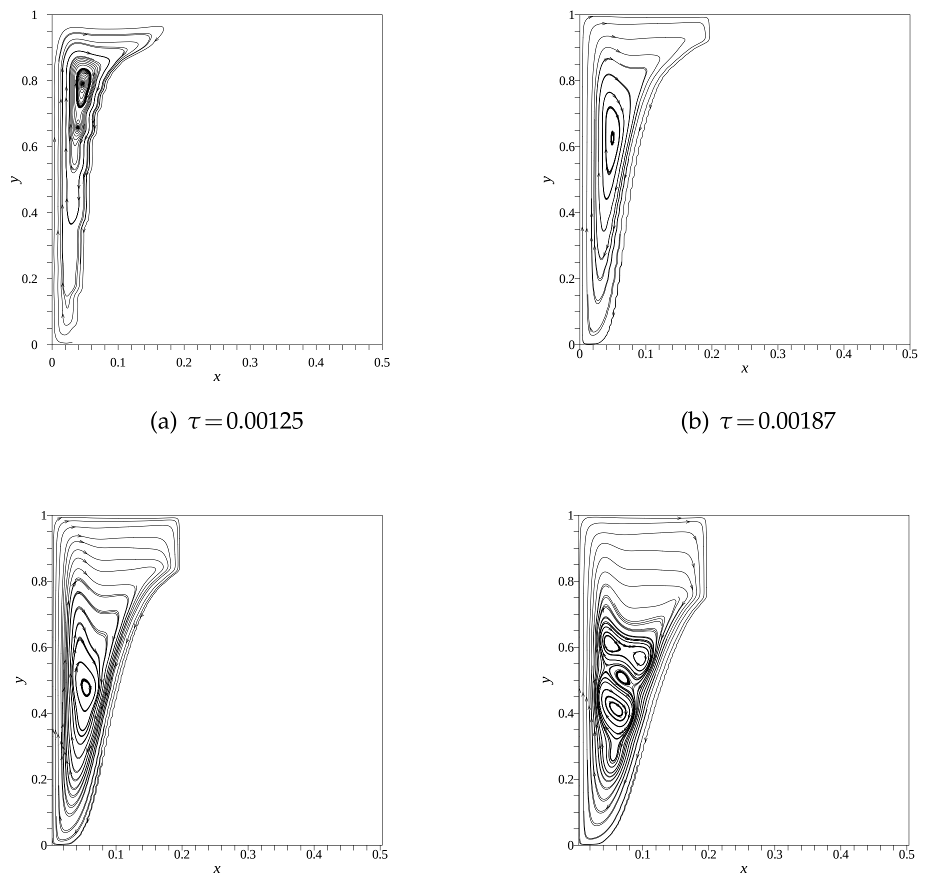
(a) $\tau = 0.00125$ (b) $\tau = 0.00187$

(c) $\tau = 0.00252$ (d) $\tau = 0.00312$

Figure 11: Temperature contours at four different τ. $Ra = 2.48 \times 10^8$, $Pr = 50$, $Ste = 0.072$.

pears, see Figs. 12(b) and 12(c). The solid-liquid interface finally reaches the cold wall and shrinks towards the right bottom corner. When the interface becomes small enough, the large amount of downward cold flow from the cold wall detaches from the solid-liquid interface. This amount of detached flow encounters the upward hot flow, which eventually causes several eddies to emerge between $y = 0.35$ and $y = 0.65$ (See Fig. 12(d)). We can also note that the isothermal lines at the corresponding positions in Fig. 11(d) become distorted due to those eddies.

From this analysis, we can find that the melting process with higher Ra number has distinct characteristics from that with lower Ra number. When the Ra number is lower than 10^7 [27], the natural convection is just a single circulation, the central part of the liquid cavity remains still. In the present simulation, since both the Ra and Pr numbers



(a) $\tau = 0.00125$

(b) $\tau = 0.00187$

(c) $\tau = 0.00252$

(d) $\tau = 0.00312$

Figure 12: Streamlines at four different τ. $Ra = 2.48 \times 10^8$, $Pr = 50$, $Ste = 0.072$.

are high, 2.48×10^8 and 50 respectively, the natural convection manifests a different flow pattern.

4.3 Performance of implementation

A Nvidia Tesla C2050 GPU was employed as platform to test the simulation efficiency. Its hardware properties were shown in the previous section (Table 2). Based on the CUDA `bandwidthTest` utility, the maximum device to device memory band width is 117.93GB/s.

Fig. 13 shows the performance of the present implementation under different lattice resolutions. The units of the left y axis are the million lattices updates per second (MLUPS). The right y axis corresponds to the throughput for device to device

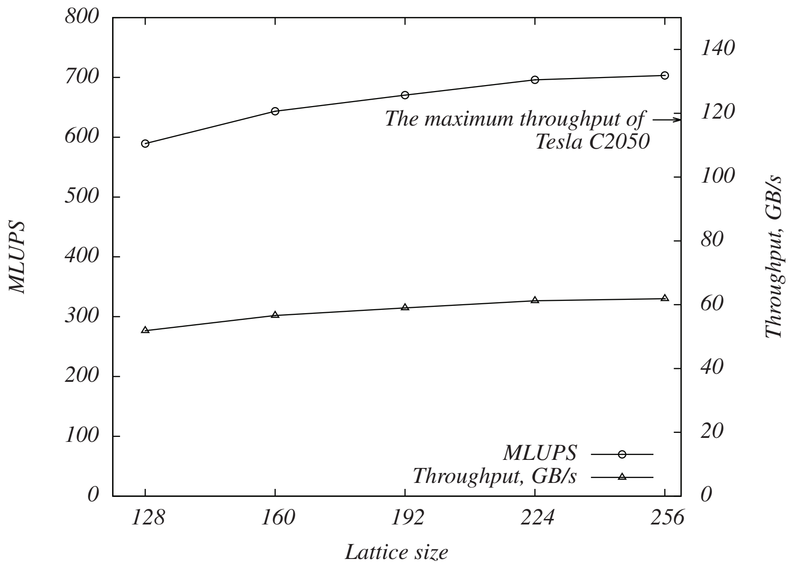


Figure 13: Performance of implementation.

throughput. The throughputs at different lattice resolutions with respect to the maximum value (117.93GB/s) range from 44.0% to 52.5%, and the simulation delivers up to 703.31 MLUPS.

5 Conclusion

In this paper, an efficient solver was designed using hybrid thermal LBM to numerically simulate the melting process. The MRT-LBM model was used to calculate the velocity field, and the enthalpy method was employed to simulate the melting phase change. This solver was optimized in order to perform on a Nvidia GPU based on the CUDA architecture (model Tesla C2050).

Experiments were also carried out to validate the numerical model and its implementation. The present simulation results are in good agreement with the experimental results, as well as the published analytic results. The temperature contours and streamlines were also presented, which showed that the melting driven by natural convection under high Ra and Pr numbers demonstrated strong convection phenomena and different characteristics from the previous published results with lower Ra and Pr numbers.

The performance tests under different grid resolutions showed that the present implementation on the Tesla C2050 GPU delivered up to 703.31 MLUPS. The device to device data throughput accounted for up to 52.5% of the hardware's maximum throughput.

The next paper will be focused on the GPU implementation on 3-dimensional melting.

Acknowledgments

This project is funded by the MICMCP 2010 project of the ANR-Stock E program of the French National Research Agency and supported by the French competitive cluster AXELERA.

References

[1] Hasnain, S. Review on sustainable thermal energy storage technologies, Part I: heat storage materials and techniques, Energy Conversion and Management, 1998, 39, 1127-1138.

[2] Burch, S. D.; Keyser, M. A.; Colucci, C. P.; Potter, T. F.; Benson, D. K. & Biel, J. P. Applications and benefits of catalytic converter thermal management. Presented at SAE Fuels & Lubricants Spring Meeting (Dearborn, MI), 1996, 961134.

[3] Gumus, M. Reducing cold-start emission from internal combustion engines by means of thermal energy storage system, Applied Thermal Engineering, 2009, 29, 652-660.

[4] Weinlader, H.; Beck, A. & Fricke, J. PCM-facade-panel for daylighting and room heating, Solar Energy, 2005, 78, 177-186.

[5] Esam M, A. Using phase change materials in window shutter to reduce the solar heat gain, Energy and Buildings, 2012, 47, 421-429.

[6] Comparison of theoretical models of phase-change and sensible heat storage for air and water-based solar heating systems, Solar Energy, 42(1989), 209-202.

[7] Bertrand, O.; Binet, B.; Combeau, H.; Couturier, S.; Delannoy, Y.; Gobin, D.; Lacroix, M.; Quéré, P. L.; Médale, M.; Mencinger, J.; Sadat, H. & Vieira, G. Melting driven by natural convection A comparison exercise: first results.

[8] Benard, C.; Gobin, D. and Zanoli, A. Moving boundary problem: heat conduction in the solid phase of a phase-change material during melting driven by natural convection in the liquid, International Journal of Heat and Mass Transfer, 1986, 29, 1669-1681.

[9] Mencinger, J. Numerical simulation of melting in two-dimensional cavity using adaptive grid, Journal of Computational Physics, 2004, 198, 243-264.

[10] Chatterjee, D. and Chakraborty, S. An enthalpy-based lattice Boltzmann model for diffusion dominated solidliquid phase transformation, Physics Letters A, 2005, 341, 320-330.

[11] Semma, E.; El Ganaoui, M.; Bennacer, R. and Mohamad, A. A. Investigation of flows in solidification by using the lattice Boltzmann method, International Journal of Thermal Sciences, 2008, 47, 201-208.

[12] Miller, W. The lattice Boltzmann method: a new tool for numerical simulation of the interaction of growth kinetics and melt flow, Journal of Crystal Growth, 2001, 230, 263-269.

[13] Miller, W.; Rasin, I. and Succi, S. Lattice Boltzmann phase-field modelling of binary-alloy solidification, Physica A: Statistical Mechanics and its Applications, 2006, 362, 78-83.

[14] Safari, H.; Rahimian, M. H. & Krafczyk, M. Extended lattice Boltzmann method for numerical simulation of thermal phase change in two-phase fluid flow, Phys. Rev. E, American Physical Society, 2013, 88, 013304.

[15] Fan, Z.; Qiu, F.; Kaufman, A. & Yoakum-Stover, S. GPU Cluster for High Performance Computing, Proceedings of the 2004 ACM/IEEE Conference on Supercomputing, IEEE Computer Society, 2004, 47.

[16] Kuznik, F.; Obrecht, C.; Rusaouen, G. and Roux, J.-J. LBM based flow simulation using GPU computing processor, Computers and Mathematics with Applications, 2010, 59, 2380-2392.

[17] Obrecht, C.; Kuznik, F.; Tourancheau, B. and Roux, J.-J. Scalable lattice Boltzmann solvers for CUDA GPU clusters, Parallel Computing, 2013, 39, 259-270.

[18] Banari, A.; Janen, C.; Grilli, S. T. & Krafczyk, M. Efficient GPGPU implementation of a lattice Boltzmann model for multiphase flows with high density ratios, Computers & Fluids, 2014, 93, 1-17.

[19] Brent, A. D.; Voller, V. R. & Reid, K. J. Enthalpy-porosity Technique For Modeling Convection-diffusion Phase Change: Application To The Melting Of A Pure Metal, Numerical Heat Transfer, 1988, 13, 297-318.

[20] Jany, P. & Bejan, A. Scaling theory of melting with natural convection in an enclosure, International Journal of Heat and Mass Transfer, 1988, 31, 1221-1235.

[21] d'Humieres, D. Multiplerelaxationtime lattice Boltzmann models in three dimensions, Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, 2002, 360, 437-451.

[22] Crouse, B.; Krafczyk, M.; Khner, S.; Rank, E. & van Treeck, C. Indoor air flow analysis based on lattice Boltzmann methods, Energy and Buildings, 2002, 34, 941-949.

[23] Van Treeck, C.; Rank, E.; Krafczyk, M.; Toelke, J. & Nachtwey, B. Extension of a hybrid thermal LBE scheme for large-eddy simulations of turbulent convective flows, Computers & Fluids, 2006, 35, 863-871.

[24] Lallemand, P. & Luo, L.-S. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability, Phys. Rev. E, American Physical Society, 2000, 61, 6546-6562.

[25] Wang, J.; Wang, D.; Lallemand, P. & Luo, L.-S. Lattice Boltzmann simulations of thermal convective flows in two dimensions, Computers & Mathematics with Applications, 2013, 65, 262-286.

[26] Contrino, D.; Lallemand, P.; Asinari, P. & Luo, L.-S. Lattice-Boltzmann simulations of the thermally driven 2D square cavity at high Rayleigh numbers, Journal of Computational Physics, 2014, 275, 257-272.

[27] Okada, M. Analysis of heat transfer during melting from a vertical wall, International Journal of Heat and Mass Transfer, 1984, 27, 2057-2066.

[28] Tolke, J. & Krafczyk, M. TeraFLOP computing on a desktop PC with GPUs for 3D CFD, International Journal of Computational Fluid Dynamics, 2008, 22, 443-456.

[29] Cuda, C. Programming guide NVIDIA Corporation, July, 2013.

[30] Obrecht, C.; Kuznik, F.; Tourancheau, B. & Roux, J.-J. The TheLMA project: A thermal lattice Boltzmann solver for the GPU, Computers &; Fluids, 2012, 54, 118-126.

[31] Mbaye, M. & Bilgen, E. Phase change process by natural convectiondiffusion in rectangular enclosures Heat and Mass Transfer, Springer-Verlag, 2001, 37, 35-42.

[32] Lim, J. & Bejan A. The Prandtl number effect on melting dominated by natural convection, Journal Name: Journal of Heat Transfer, (Transactions of the ASME (American Society of Mechanical Engineers), Series C), 1992.

[33] Voller, V. & Prakash, C. A fixed grid numerical modelling methodology for convection-diffusion mushy region phase-change problems, International Journal of Heat and Mass Transfer, 1987, 30, 1709-1719.

[34] J. Miranda Fuents, Développement dun modèle de Boltzmann sur gaz réseau pour létude du changement de phase en présence de convection naturelle et de rayonnement, PhD. Thesis, 2013.

[35] He, X. & Luo, L.-S. A priori derivation of the lattice Boltzmann equation, Phys. Rev. E, American Physical Society, 1997, 55.

[36] Lallemand, P. & Luo, L.-S. Theory of the lattice Boltzmann method: Acoustic and thermal properties in two and three dimensions, Phys. Rev. E, American Physical Society, 2003, 68, 036706.

[37] Chen, Y.; Ohashi, H. & Akiyama, M. Prandtl number of lattice BhatnagarGrossKrook fluid, Physics of Fluids (1994-present), 1995, 7, 2280-2282.