A dual-lattice hydrodynamic-thermal MRT-LBM model implemented on GPU for DNS calculations of turbulent thermal flows

Received 10 June 2022
Revised 16 September 2022
3 October 2022
Accepted 18 October 2022

T.O.M. Forslund, I.A.S. Larsson, J.G.I. Hellström and T.S. Lundström
Department of Engineering Sciences and Mathematics, Division of Fluid and Experimental Mechanics, Luleå University of Technology, Luleå, Sweden

Abstract

Purpose – The purpose of this paper is to present a fast and bare bones implementation of a numerical method for quickly simulating turbulent thermal flows on GPUs. The work also validates earlier research showing that the lattice Boltzmann method (LBM) method is suitable for complex thermal flows.

Design/methodology/approach – A dual lattice hydrodynamic (D3Q27) thermal (D3Q7) multiple-relaxation time LBM model capable of thermal DNS calculations is implemented in CUDA.

Findings – The model has the same computational performance compared to earlier publications of similar LBM solvers. The solver is validated against three benchmark cases for turbulent thermal flow with available data and is shown to be in excellent agreement.

Originality/value – The combination of a D3Q27 and D3Q7 stencil for a multiple relaxation time -LBM has, to the authors' knowledge, not been used for simulations of thermal flows. The code is made available in a public repository under a free license.

Keywords Lattice Boltzmann method, Turbulence, Thermal flows, Direct numerical simulation

Paper type Research paper

1. Introduction

The lattice Boltzmann method (LBM) is a cellular automata method developed as a more stable and computationally effective version of the Lattice Gas Automata (LGA) method. The LGA method, as described in Frisch *et al.* (1986), tracks discrete particles through a collection of nodes with static connections. When two particles are in the same node at the

This work was made possible by funding from the Swedish Research Council (Grant No. 2017-04390).

In the interest of transparency, data sharing and reproducibility, the author(s) of this article have made the data underlying their research openly available. It can be accessed by following the link here: https://gitlab.com/c8383/thermal-mrt-lbm

International Journal of Numerical Methods for Heat & Fluid Flow
Vol. 33 No. 5, 2023
pp. 1703-1725
Emerald Publishing Limited
0961-5539
DOI 10.1108/HFF-06-2022-0339

same time a collision step is carried out and the design of this collision operator impacts the dynamic behavior of the "gas." In contrast, the LBM does not track individual particles and instead uses ensemble averages in a control volume as quantities of interest. Because of this, the method is often referred to as a mesoscopic method (Chen and Doolen, 1998; Li *et al.*, 2016), where every control volume holds the velocity information of the particles contained in it, i.e. a discretized version of the local velocity distribution. Recently, the method has gained increased interest due to its favorable parallelizability properties, especially for implementations on GPUs. As demonstrated in Delbosc (2015) the expected reduction in calculation time is approximately two orders of magnitude for a GPU implementation as compared to a CPU implementation. Usage of GPUs thus enables fast DNS and LES simulations of complex fluid phenomena and is especially well suited for problems with complex geometries and evenly distributed local Reynolds numbers, such as porous media flows (Gao *et al.*, 2015).

LBM models capable of accurately resolving turbulent structures have been reported by Suga *et al.* (2015), Geier *et al.* (2006), Premnath and Banerjee (2011) and Abdul-Kadhim *et al.* (2019) among others, and as shown in Delbosc *et al.* (2014), Delbosc (2015) and Li *et al.* (2016) hydrodynamic models can be combined with a convective-diffusive LBM model to model thermal fluids. Therefore, a combination of these models is assumed to yield accurate models of resolved turbulent thermal fluid phenomena. In this article, a GPU implementation of an LBM model which is stable for low viscosities and thermal diffusivities is reported in detail. The model is tested against three cases for which validation data is available. Excellent agreement is obtained.

2. Theory

2.1 Lattice Boltzmann methods

The evolution of a gas is governed by the Boltzmann equation and it is from this equation that the LBM is derived according to:

$$\frac{\partial f}{\partial t} + c_j \frac{\partial f}{\partial x_j} = \Omega(f). \tag{1}$$

Here, f is the distribution function which, in general, is a function of space (x_i), time (t), particle velocity (c_j) and Ω is a collision operator. If the velocity distribution is discretized, the set of values in it can be written as a vector f_i. The vector c_j is then a discrete set of velocity vectors called e_{ij}. Usually the procedure is separated into two steps, a *collision step* and a *streaming step*. With a discretized version of the distribution function we have the following set of equations:

$$f_i(x_j, t) = \tilde{f}_i(x_j, t) + \Omega_i(x_j, t), \tag{2}$$

$$\tilde{f}_i(x_j + e_{ij}, t + 1) = f_i(x_j, t). \tag{3}$$

Here e_{ij} is, as mentioned earlier, called the discretized velocity vector which depends on how the spatial discretization of the distribution is done and notice that equation (2) is the collision step and equation (3) is the streaming step. Also notice that f_i is the discretized distribution function after being relaxed while $\tilde{f}_i$ is the distribution before relaxation. The

control volume average density (ρ) and velocity (u_i) can be obtained from the distribution function by the relations (Chen and Doolen, 1998):

$$\rho = \sum_i f_i, \tag{4}$$

$$u_i = (1/\rho)\sum_i f_i e_{ij}. \tag{5}$$

These quantities are always *preserved* during the collision step which follows naturally from continuity and momentum preservation. A natural assumption for the velocity distribution of a gas within an isolated control volume is that it should tend toward the Boltzmann distribution as time goes to infinity. This assumption was first used by Bhatnagar *et al.* (1954). Because discrete time steps are being taken, a relaxation parameter (τ) is assigned which decides how quickly the control volume approaches the equilibrium state. The collision operator can then be written as:

$$\Omega_i(f_i) = f_i + \left(f_i^{eq} - f_i\right)\frac{1}{\tau}. \tag{6}$$

Here, τ is a relaxation parameter being related to the diffusivity of the gas, i.e. the viscosity for the hydrodynamic equations and thermal diffusivity for the diffusion equation. The term f_i^{eq} is the *equilibrium distribution*, the form of which decides the set of equations that are solved. These models are referred to as single relaxation time (SRT) models.

2.2 Moments

Moments are constructed from the discretized velocity vector and they have a physical interpretation such as momentum or density. In general, the moment $\mu_{x^n y^m z^o}$ can be written as (Premnath and Banerjee, 2009):

$$\mu_{x^n y^m z^o} = \sum_i f_i (e_{ix})^n (e_{iy})^m (e_{iz})^o \tag{7}$$

where the summation convention is not used for the index j. When the moment operator acts on a distribution f_i the desired physical quantity is calculated. For example, $f_i e_{ix}$ gives the x-direction momentum.

2.3 Forms of the equilibrium distribution

The LBM can be used to solve many different coupled PDEs depending on the choice of equilibrium distribution. Some more exotic examples include the shallow water equations (Tubbs and Tsai, 2019) and economic simulations of income distribution (Cerdá *et al.*, 2013). In this work, the forms of the equilibrium distribution will be chosen so that the Navier–Stokes equations for the hydrodynamic modeling and the advection-diffusion equation for the thermal modeling can be solved.

2.3.1 Bhatnagar–Gross–Krook SRT. In the Bhatnagar–Gross–Krook (BGK) model, the vector f_i^{eq} represents the equilibrium state given by the Boltzmann distribution and because exponentials are computationally expensive to evaluate, Taylor expansion version is used (Qian *et al.*, 1992; Chen and Doolen, 1998):

$$f_i^{eq} = \rho w_i \left(1 + 3e_{ij}u_j + 4.5(e_{ij}u_j)^2 - 1.5u_j u_j\right). \tag{8}$$

In this case, the summation convention is applied over the index j but not i. Also, w_i are weights and details regarding how to derive these quantities can be found in Kruger *et al.* (2017). This version of f_i^{eq} is only valid when the lattice velocity c is equal to 1, i.e. natural units are used. The BGK-SRT collision model has drawbacks as compared to the multiple relaxation time (MRT) and cascaded types because it has stability issues and is consistently outperformed in all areas (D'Humières *et al.*, 2002). Although the process by which the relaxation is carried out differs between the MRT and SRT models the equilibrium function is usually the same as equation (8).

2.3.2 Thermal equilibrium distribution. The thermal equilibrium distribution can be written in the following form (Li *et al.*, 2016; Yoshida and Nagaoka, 2010; Delbosc *et al.*, 2014):

$$f_i^{eq} = Tw_i(1 + \epsilon u_j e_{ij}). \tag{9}$$

In equation (9), T is the temperature of the lattice point defined analogously to the density ρ, see equation (4) as:

$$T = \sum_i f_i. \tag{10}$$

u_j is the local velocity in coordinate direction j, this velocity can either be specified or obtained from a hydrodynamic solver. The variable ϵ depends on the choice of lattice and for the case of a D3Q7 lattice as used in this work the value is $\epsilon = 4$ while for a D3Q6 lattice $\epsilon = 2$.

2.4 Multiple relaxation time

The BGK-LBM and MRT-LBM collision operator, see equations (2) and (3), can in its general form be written as:

$$\Omega_i(f_j) = -(M_{ik})^{-1}\hat{S}_{ki}\left[M_{ij}f_j - M_{ij}f_j^{eq}\right] + F_i. \tag{11}$$

Here, M_{ij} is the moment matrix, $\hat{S}_{kj}$ is a diagonal matrix called the relaxation coefficient matrix and F_i is some general forcing term. For the special case of a BGK model the relaxation coefficient matrix is the identity matrix times a constant. F_i is a general source term which, in general, can be a function of f_i depending on the forcing scheme. Changing the values of the relaxation matrix affects the numerical stability. This general form holds true for any MRT-LBM model but the choice of M_{ij}, $\hat{S}_{ki}$ and f_j^{eq} will affect the set of equations being solved and the numerical stability.

2.5 Boundary conditions

Two types of boundary conditions are applied in this work, *periodic boundary conditions* and *bounce-back boundary conditions*. The periodic boundary condition connects the domain along a specified axis as illustrated in Figure 1. The application of the boundary condition needs to be made with proper consideration for the structures present in the physical system that is being modeled. This is because the condition acts as a box filter for the large-scale

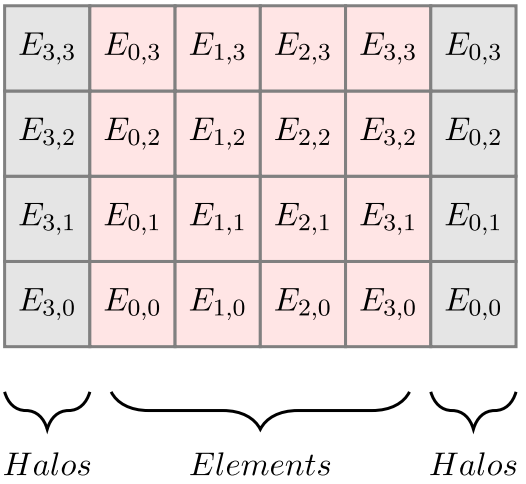


Figure 1.
Visualization of the periodic boundary condition, the nodes wrap around at the edges

structures and may remove some physically important processes if not set up in a proper way (Forslund *et al.*, 2021). For the validation cases presented, the same domains as the reference cases are used.

The bounce-back boundary condition is a second-order accurate no-slip wall condition (Mohamad, 2011; Bouzidi *et al.*, 2001; Ginzburg and D'Humières, 2003) which inverts all the velocities going into the wall which is then streamed back during the next time-step. The following operation is applied to the wall element at each time-step:

$$f_B(c_i) = f(-c_i). \tag{12}$$

For the implementation, a discretized velocity vector e_{ij} is chosen in such a way that an inversion of the vector along the element axis i corresponds to the reflection operation specified by equation (12). An effect of the bounce-back boundary condition is that the wall distance for the elements closest to the wall is half an element side-length (Mohamad, 2011).

2.6 Emergent field equations

In this work, the construction of the collision operator $\Omega(f_i)$ will be designed in such a way that two different types of partial differential equations are solved for. These are the hydrodynamical equation which describes an incompressible fluid, sometimes called the Navier–Stokes equations:

$$\frac{du_i}{dt} + u_j u_{i,j} = -p_{,i} + \nu u_{i,jj} + a_i, \tag{13}$$

$$u_{i,i} = 0. \tag{14}$$

where u_i is the velocity, p is the modified pressure which is equal to the physical pressure times density, a_i is a source term and ν is the kinematic viscosity. For the thermal part the collision operator is adapted so that the convection-diffusion equation is solved for:

$$\frac{d\phi}{dt} + u_j \phi_{,j} = \alpha \phi_{,jj} + \phi_s. \tag{15}$$

Here, ϕ can be any variable which are convected by the flow, ϕ_s is a source term and α is the diffusion coefficient. For the case of a thermal field, $\phi = T$ and α is referred to as the *thermal diffusion*.

3. Method

In this section, the implementation of the thermal and hydrodynamic model is described in detail.

3.1 MRT-D3Q27 hydrodynamic model

For a 3D gas lattice automata there are many possible choices of lattice, these include the D3Q7, D3Q13 and D3Q19 among others (Mohamad, 2011; Geier *et al.*, 2006). In this notation, a *DnQm* lattice would have *n* spatial dimensions and *m* number of distributions. As discussed in Geier *et al.* (2006), popular velocity sets such as those mentioned in the preceding sentence are insufficient to adjust all the relevant moments independently, for example, the μ_{xyz} moment which is important for the short wavelength flow structures by its impact on the shear rate advection. Because properly resolving turbulent structures are of interest for the hydrodynamic part, a D3Q27 lattice set is used, the basis vectors are defined by:

$$e_{i0} = (j\%3) - 1 \tag{16}$$

$$e_{i1} = (\lfloor j/3 \rfloor \%3) - 1 \tag{17}$$

$$e_{i2} = (\lfloor j/9 \rfloor) - 1. \tag{18}$$

In these equations $\lfloor\rfloor$ is the flooring operator and % is the modulo operator, the subscript index 0, 1 and 2 is the coordinate direction which are also sometimes referred to as e_{ix}, e_{iy} and e_{iz}. Written out explicitly e_{ij} becomes:

$$e_{ij} = \begin{Bmatrix} -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 \\ -1 & -1 & -1 & 0 & 0 & 0 & 1 & 1 & 1 & -1 & -1 & -1 & 0 & 0 & 0 & 1 & 1 & 1 & -1 & -1 & -1 & 0 & 0 & 0 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{Bmatrix}. \tag{19}$$

This ordering differs as compared to most works and is useful because the wall bounce-back condition, in this formulation, is equivalent to a reversal of the streaming operator. The D3Q27-distribution is visualized in Figure 2, where red are length 1, blue are length $\sqrt{2}$ and green are $\sqrt{3}$; Each link corresponds to a different weight, there are four distinct weights these are ω_0 = 8/27 (grey), ω_1 = 2/27 (red), ω_2 = 1/54 (blue) and ω_3 = 1/216 (green). Written out the weight vector w_i becomes:

$$w_i = \{\omega_3, \omega_2, \omega_3, \omega_2, \omega_1, \omega_2, \omega_3, \omega_2, \omega_3, \omega_2, \omega_1, \omega_2, \omega_1, \omega_0, \omega_1, \omega_2,$$
$$\omega_1, \omega_2, \omega_3, \omega_2, \omega_3, \omega_2, \omega_1, \omega_2, \omega_3, \omega_2, \omega_3\} \tag{20}$$

In this work, the same moment basis as that presented in Premnath and Banerjee (2011) is used. In terms of the vector set the following orthogonal matrix basis is applied:

$$
\begin{aligned}
M_{ij} = [&e_{ix}^0,\\
&e_{ix},\\
&e_{iy},\\
&e_{iz},\\
&e_{ix}e_{iy},\\
&e_{ix}e_{iz},\\
&e_{iy}e_{iz},\\
&e_{ix}e_{ix}-e_{iy}e_{iy},\\
&(e_{ix}e_{ix}+e_{iy}e_{iy}+e_{iz}e_{iz})-3e_{iz}e_{iz},\\
&(e_{ix}e_{ix}+e_{iy}e_{iy}+e_{iz}e_{iz})-2e_{iz}^0,\\
&3e_{ix}(e_{iy}e_{iy}+e_{iz}e_{iz})-4e_{ix},\\
&3e_{iy}(e_{ix}e_{ix}+e_{iz}e_{iz})-4e_{iy},\\
&3e_{iz}(e_{ix}e_{ix}+e_{iy}e_{iy})-4e_{iz},\\
&e_{ix}(e_{iy}e_{iy}-e_{iz}e_{iz}),\\
&e_{iy}(e_{ix}e_{ix}-e_{iz}e_{iz}),\\
&e_{iz}(e_{ix}e_{ix}-e_{iy}e_{iy}),\\
&e_{ix}e_{iy}e_{iz},\\
&3(e_{ix}e_{ix}e_{iy}e_{iy}+e_{ix}e_{ix}e_{iz}e_{iz}+e_{iy}e_{iy}e_{iz}e_{iz})-4(e_{ix}e_{ix}+e_{iy}e_{iy}+e_{iz}e_{iz})+4e_{ix}^0,\\
&3(e_{ix}e_{ix}e_{iy}e_{iy}+e_{ix}e_{ix}e_{iz}e_{iz}-2e_{iy}e_{iy}e_{iz}e_{iz})-2(2e_{ix}e_{ix}-e_{iy}e_{iy}-e_{iz}e_{iz}),\\
&3(e_{ix}e_{ix}e_{iy}e_{iy}-e_{ix}e_{ix}e_{iz}e_{iz})-2(e_{iy}e_{iy}-e_{iz}e_{iz}),\\
&3(e_{ix}e_{ix}e_{iy}e_{iz})-2e_{iy}e_{iz},\\
&3(e_{ix}e_{iy}e_{iy}e_{iz})-2e_{ix}e_{iz},\\
&3(e_{ix}e_{iy}e_{iz}e_{iz})-2e_{ix}e_{iy},\\
&9(e_{ix}e_{iy}e_{iy}e_{iz}e_{iz})-6(e_{ix}e_{iy}e_{iy}+e_{ix}e_{iz}e_{iz})+4e_{ix},\\
&9(e_{ix}e_{ix}e_{iy}e_{iz}e_{iz})-6(e_{ix}e_{ix}e_{iy}+e_{iy}e_{iz}e_{iz})+4e_{iy},\\
&9(e_{ix}e_{ix}e_{iy}e_{iy}e_{iz})-6(e_{ix}e_{ix}e_{iz}+e_{iy}e_{iy}e_{iz})+4e_{iz},\\
&27(e_{ix}e_{ix}e_{iy}e_{iy}e_{iz}e_{iz})-18(e_{ix}e_{ix}e_{iy}e_{iy}+e_{ix}e_{ix}e_{iz}e_{iz}+e_{iy}e_{iy}e_{iz}e_{iz})+12(e_{ix}e_{ix}+e_{iy}e_{iy}+e_{iz}e_{iz})-8e_{ix}^0].
\end{aligned}
\tag{21}
$$

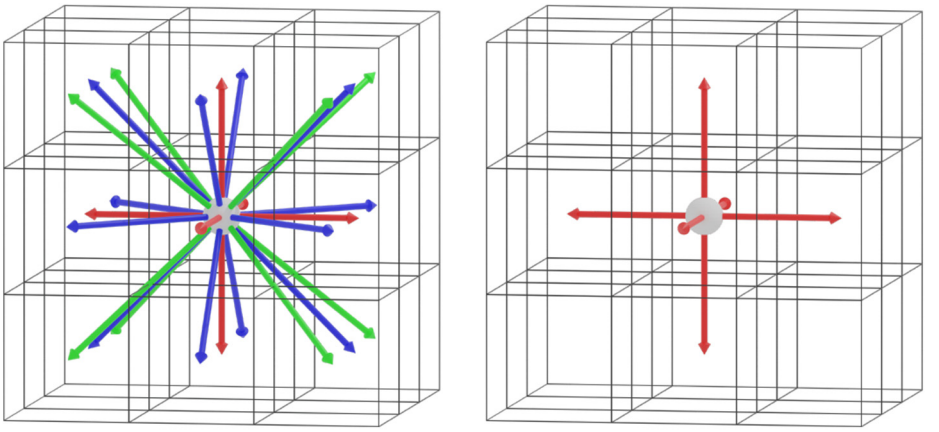


Figure 2.
Visualization of D3Q27 (left) and D3Q7 (right) lattices, the central node is indicated by the grey sphere

Notes: Red are length 1, blue are length $\sqrt{2}$ and $\sqrt{3}$ green are; Each link corresponds to a different weight, there are four distinct weights: $\omega_0 = 8/27$ (grey), $\omega_1 = 2/27$ (red), $\omega_2 = 1/54$ (blue) and $\omega_3 = 1/216$ (green)

In this equation, the same index indicates element-wise multiplication instead of the more common summation convention. Although a pair-wise orthogonal moment basis is chosen for this model it may be noted that the moment basis does not need to be orthogonal to obtain good numerical stability as shown by Fei *et al.* (2019). The choice of relaxation factors in the matrix is key for the stability of the computations. As noted by the solver implemented in Suga *et al.* (2015) more relaxation is required for the energy terms. Ideally it is preferred that all relaxation terms except for those related to moment, density and viscosity should be exactly 2, as to minimize the impact of the moment relaxation on the solution. Guided by this and a limited parameter sweep, the following values for the relaxation matrix are used:

$$\hat{S}_{kj} = \{0,0,0,0,s_\nu,s_\nu,s_\nu,s_\nu,s_\nu,1.54,1.5,1.5,1.5,1.83,1.83,1.83,1.74,1.45,1.99,\\ 1.99,1.99,1.99,1.99,1.74,1.74,1.74,1.74\}. \tag{22}$$

The relaxation factors for the preserved moments, i.e. physical quantities that does not change dues to conservation laws (density and momentum), are set to zero. Moment 4–8 relates to the viscosity tensor and is changed to adapt the viscosity. The rest of the relaxation factors are set to a value which promotes stability. The relation between the viscosity and relaxation factors for the moments s_ν can be derived via a Chapman–Enskog analysis. This has been done before in, for example, Premnath *et al.* (2009), and the following relation between the relaxation and viscosity holds true for the D3Q27 lattice:

$$\nu = \frac{1}{3}\left(\frac{1}{s_\nu} - \frac{1}{2}\right). \tag{23}$$

The forcing scheme used for the body force is first-order and is applied to the postcollision distribution function, hence the forcing term is described by the expression (Suga *et al.*, 2015):

$$F_j = \omega_j \rho \frac{e_{ij} a_i}{c_s^2}. \tag{24}$$

where summation is not carried out across the j index and a_i is the fluid acceleration in coordinate direction j.

3.2 MRT-D3Q7 thermal lattice

The model implemented for the thermal D3Q7-model is described in Yoshida and Nagaoka (2010) and the basis vectors are defined by:

$$e_{ij} = \begin{Bmatrix} 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{Bmatrix}. \tag{25}$$

A visualization of the distributions can be seen to the right in Figure 2. Each link corresponds to a different weight, there are two distinct weights which are $\omega_0 = 1/4$ (grey in the middle), $\omega_1 = 1/8$ (red). Written out the weight vector w_i becomes:

$$w_i = \{\omega_1, \omega_1, \omega_1, \omega_0, \omega_1, \omega_1, \omega_1\} \tag{26}$$

The choice of moment basis results in the following moment matrix:

$$M_{ij} = \left[e_{ix}^0, e_{ix}, e_{iy}, e_{iz}, 6 - 7(e_{ij})^2, 3e_{ix}^2 - (e_{ij})^2, e_{iy}^2 - e_{iz}^2\right]. \tag{27}$$

In this equation, $(e_{ij})^2$ is the square of the norm of the discretized velocity distribution for link i. The relaxation matrix $\hat{S_{kj}}$ for this thermal model can contain additional off-diagonal terms dependent on whether the diffusion tensor is anisotropic or not. Because only fully isotropic media will be considered in this work, the following form of the relaxation matrix is obtained:

$$\hat{S_{kj}} = \{0, s_\alpha, s_\alpha, s_\alpha, s_4, s_5, s_6\}. \tag{28}$$

Here, $s_0 = 0$ because the value of it does not affect the numerical solution at all and it can be omitted. The factors s_4, s_5 and s_6 do not affect the leading-order approximation but impact the error terms. For the CUDA implementation, these terms are kept as compile-time constants that can be adjusted for accuracy and stability. For the D3Q7 lattice, a Chapman–Enskog analysis results in a different relation between the thermal diffusivity α and the corresponding relaxation factors s_α related to it according to:

$$\alpha = \frac{1}{4}\left(\frac{1}{s_\alpha} - \frac{1}{2}\right). \tag{29}$$

The numerical speed of sound c_s differs between the D3Q27 and the D3Q7 model but as noted by Hajabdollahi and Premnath (2018) the models can still be used in conjunction. The scheme used for the thermal source is first-order and is applied to the postcollision distribution function, the source term described by the expression:

$$S_j = \omega_j S dt. \tag{30}$$

where S is the source term.

3.3 Two-way coupling using the Boussinesq approximation

When simulating fluid dynamical systems where the density varies with temperature to the extent that the liquid cannot be treated as isothermal, the Boussinesq approximation can be used, given that the density difference is sufficiently small. A model for handling non-isothermal fluids is necessary if natural convection influence the flow. The approximation assumes that the expansion related to compressibility is negligible, therefore the density differences impact on the dynamical behavior can be approximated as a difference in the gravitational acceleration acting on the fluid elements, i.e.:

$$a_i = g_i \beta T \tag{31}$$

where a_i is the acceleration vector acting on the fluid element, g_i is the gravitational acceleration, β is a dimensional coupling constant and T is the temperature. For numerical stability, the variation of T should be kept around zero to avoid large pressure gradients in the simulation domain.

3.4 Implementation in CUDA framework

There are some considerations that need to be made, regarding memory alignment and access, to enable optimal performance of the code. The computational domain consists of an equirectangular lattice of cubic elements where each element holds the data presented in Table 1.

In Table 1, f_i is the double-buffered D3Q27 and D3Q7 distribution for the hydrodynamic and thermal LBM solvers, respectively, u_i is the velocity vector, ρ is the density, T is the temperature and the wall character is a flag that is used to mark whether the current cell is a wall cell or not. The solver is implemented so that 64-bit floating point accuracy can be used instead of 32-bit, but this leads to significantly longer calculation times on most GPU architectures since they are not optimized for double precision. Because the Mach number is low, it is not important to save the velocities in each time-step, therefore a saving interval is set to some specified number of time-steps to save memory bandwidth in the main loop.

3.5 Memory arrangement and streaming

As indicated in Delbosc *et al.* (2014), Delbosc (2015) memory alignment, when the memory is transferred to the kernels, is important for optimal performance. Therefore, the memory is arranged as indicated in Figure 3. The memory arrangement, called coalesced memory, is necessary for the transfers to be aligned and therefore enabling the usage of the entire

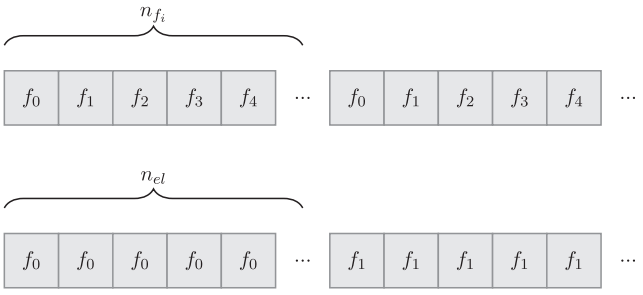
Table 1. Memory allocated for each cell

Name	Amount	Type	Size (bytes)
f_i	34×2	float	272
u_i	3	float	12
ρ	1	float	4
T	1	float	4
wall	1	char	1
total	×	×	293

memory bus. This, however, introduces a problem because the streaming step is necessarily noncoalesced for this memory arrangement. It was concluded in Delbosc *et al.* (2014) that reading in values from the surrounding nodes before executing the collision kernel is faster than saving to the surrounding nodes after the collision calculations. This is due to uncoalesced reads being faster than uncoalesced writes. Therefore, the streaming step is carried out at the beginning of the combined collision-streaming kernel.

3.6 Overall program structure

The overall program structure consists of an initialization stage and three main loops (see Figure 4). The inner loop updates the distributions using the MRT-LBM algorithm. After $n - 1$ operations a special version of the kernel is called that saves monitored variable data



Note: n_{fi} is the amount of distributions, for our case 34, and n_{el} is the amount of elements in the simulation

Figure 3.
Figure of standard memory arrangement (top) and arrangement for coalesced access bottom

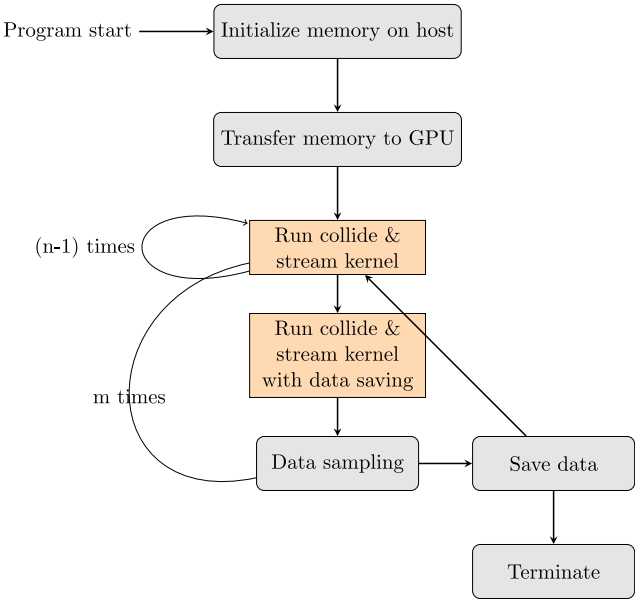


Figure 4.
Program structure flowchart

to GPU memory. The data is then sampled and transferred to the host memory. Once the outer sweep is carried out m times, data is saved to the hard drive and the program either terminates or carries out another outer sweep. This loop structure is designed to avoid the progressively slower bottlenecks of host RAM memory and hard drive transfer speeds by minimizing the amount of transfer calls. With a proper choice of the intervals n and m, the impact of saving data on simulation speed is negligible.

3.7 Stream and collision kernel structure

The stream and collision kernel is divided into four steps, the streaming step which reads in the surrounding distributions, the fluid collision step which applies equation (11) to the D3Q27 model, the thermal collision step which applies the same equation to the D3Q7 model and finally the saving step which saves all relevant variables to the GPU memory.

3.8 OpenGL integration

Because the simulation memory is stored on the GPU it is possible to use the memory to render the flow fields directly by using graphical libraries such as Open Graphics Library (OpenGL) or DirectX. The solver presented in this article can be coupled to an OpenGL live renderer by defining the memory/variables of interest for rendering as a *glBufferData()* object. This gives the ability to both CUDA and OpenGL to access the memory of the variables of interest such as density, velocity and temperature. Video examples of live rendering are provided as appended additional material to the article.

3.9 Code availability

The code is open source and available for download from the repository https://gitlab.com/c8383/thermal-mrt-lbm

4. Simulation setup

The solver is validated by comparison to three thermal DNS test cases, the plane channel flow case as presented in Kawamura *et al.* (1998), a porous media test case as presented in Chu *et al.* (2019) and a Rayleigh–Bénard convection case compared to the data presented in Wörner (1994). The two-way coupling between the thermal and hydrodynamic lattice is only applied to the Rayleigh-Bénard case.

4.1 Plane channel

Plane channel flow is a standard case for validating CFD codes and is also of academic interest due to the flow properties dependence on the nature of near-wall turbulence. Because of this, there are many sources of reliable DNS data for both the turbulent and thermal statistics of this case, some examples are: Alcántara-Ávila *et al.* (2021) and Kawamura *et al.* (1998). For the thermal plane channel flow case there are walls at the top and bottom where a constant temperature and no-slip condition are applied. On the other walls periodic boundary conditions are used, see Figure 5. The temperature source is varied so that the average temperature of the domain is equal to 100 (natural units). The flow is driven by a constant pressure gradient $F_x(t)$ which is set to a value so that the time-average Re_τ is constant. The mesh consists of cubic elements of unit size in a lattice of size $nX \times nY \times nZ = 192 \times 288 \times 576$.

There are three Reynolds numbers that are commonly used to characterize the dynamics of the flow for this case, these are:

$$Re = \frac{\overline{U}L}{\nu}, \tag{32}$$

$$Re_0 = \frac{U_0\delta}{\nu}, \tag{33}$$

$$Re_\tau = \frac{u_\tau\delta}{\nu}. \tag{34}$$

where Re is based on the mean channel stream-wise velocity $\overline{U}$, the height of the channel L and kinematic viscosity ν. Re_0 uses the channel central velocity U_0 and channel half height $\delta = L/2$. Re_τ is called the *friction Reynolds number* and is based on the *friction velocity* which is defined as a function of the wall shear as:

$$u_\tau \equiv \sqrt{\frac{\tau_w}{\rho}} \approx \sqrt{\tau_w} \tag{35}$$

where the last approximation can be made because ρ is kept at approximately 1 across the flow field. The wall shear stress τ_w is defined as:

$$\tau_w = \rho\nu\left(\frac{d < U >}{dz}\right)_{z=0} \approx \nu\left(\frac{d < U >}{dz}\right)_{z=0} \tag{36}$$

where z is the wall normal coordinate and $z = 0$ is at the wall. From these quantities the wall shear distance z^+ and wall shear velocity u^+ can be defined as:

$$z^+ = \frac{u_\tau z}{\nu} \tag{37}$$

$$u^+ = \frac{u}{u_\tau} \tag{38}$$

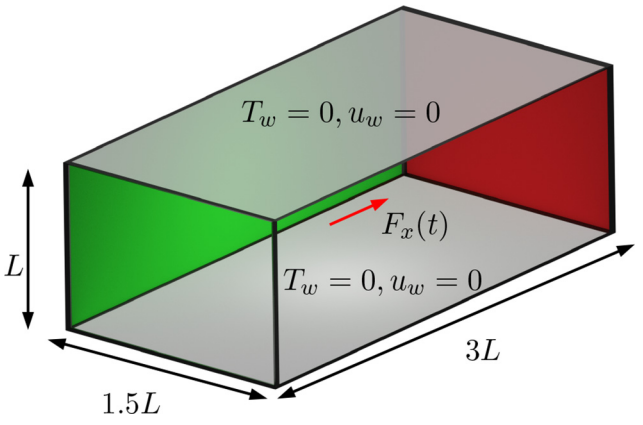


Figure 5.
Geometry and boundary conditions for plane channel flow, red and green are periodic in x (red) and y (green) coordinate directions

The dimensionless number which characterizes the thermal dynamics is the Prandtl number, which is given as:

$$Pr = \frac{\nu}{\alpha}. \tag{39}$$

where α is the thermal diffusivity. The numerical values used for the simulation are:

$$\rho_0 = 1$$
$$Pr = 1.0$$
$$\nu = 0.0006$$
$$\overline{Re_\tau} = 180$$
$$\overline{T} = 100$$
$$L = 191.$$

Natural units are used for all values and are therefore dimensionless. The initial velocity distribution is a completely uniform x-direction velocity profile with added numerical noise. The initial temperature profile is a uniform profile of $T = 100$. The flow field is allowed to develop for 2,500,000 time-steps which is equivalent to approximately 80 wash-outs before data sampling begins. The data is sampled over the time-span of 2,500,000 time-steps with samples being taken every 40th time-step. Complete field data of standard deviations and mean values are exported every 100,000 time-steps, or approximately 3.5 wash-outs.

4.2 Porous media

The thermal porous media flow is validated against one of the cases presented in Chu *et al.* (2019). The geometry consists of cylinders with a quadratic cross section placed in a staggered arrangement, see Figure 6. The flow and thermal dynamics in the porous media are characterized by two dimensionless numbers, these are Re and the Prandtl number Pr, Re is here defined as:

$$Re = \frac{U_{Darcy} d_c}{\nu} \tag{40}$$

where U_{Darcy} is the stream-wise velocity averaged over both the solid and liquid phases, d_c is the square rod side-length and intrarod distance as illustrated in Figure 6. α is the thermal diffusivity. The flow is driven by a varying force $F_x(t)$ which enforces a constant Darcy velocity, and a source term for the temperature keeps a constant average temperature $\overline{T}$ of 1 across the entire domain. The mesh consists of cubic elements of unit size in a lattice of size $nX \times nY \times nZ = 256 \times 256 \times 256$. The numerical values used for the simulation are:

$$Pr = 1.0$$
$$\nu = 0.00192$$
$$U_{Darcy} = 0.015$$
$$\overline{T} = 1$$
$$d_c = 64$$

The initial velocity is zero for all components in the whole domain and the average temperature is set to 1 for all fluid elements. The flow is allowed to initialize for 6,000,000

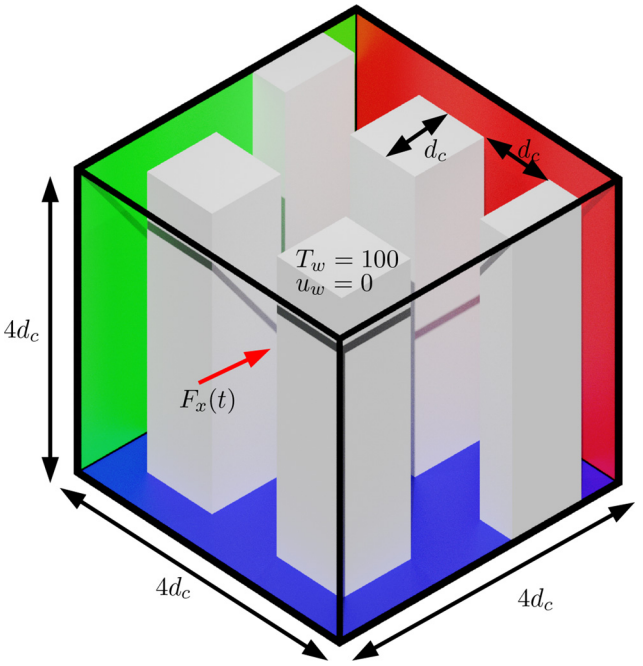


Figure 6.
Geometry and boundary conditions for porous media flow, red and green are periodic in x (red), y (green) and z (blue) coordinate directions

time-steps, or approximately 460 wash-outs before sampling begins. Values are sampled every 40th time-step. Complete field data of standard deviations and mean values are exported every 400,000 time-steps, or approximately 30 wash-outs.

4.3 Rayleigh–Bénard convection

For Rayleigh–Bénard convection there are two relevant dimensionless numbers (Wörner, 1994; Yilmaz, 2020), these are usually chosen to Pr, see equation (39), and the Rayleigh number Ra defined as:

$$Ra = \frac{g\beta\Delta T L^3}{\nu\alpha} \tag{41}$$

and using natural units $g = 1$, $\Delta T = 1$ Ra can be written as:

$$Ra = \frac{\beta L^3}{\nu\alpha}. \tag{42}$$

The geometry and boundary condition of the case is presented in Figure 7 (Wörner, 1994). The flow is driven by the acceleration g and temperature difference between the top wall $T_w = 0$ and bottom wall $T_w = 1$. The domain is periodic in the x and y directions with the side length to height ratio being 8. The mesh consists of cubic elements of unit size in a lattice of size $nX \times nY \times nZ = 512 \times 512 \times 64$. The values used for the variables in the simulation are:

$$\beta = 0.00005$$
$$\alpha = 0.005161$$
$$Pr = 0.71$$
$$L = 62.$$

These simulation values yield $Pr = 0.71$ and, as a result, $Ra = 630 \times 10^3$. The velocity field is initially set to zero everywhere with numerical noise added, the thermal field is initialized as $T = 0.5$ for all fluid elements. The flow is allowed to stabilize for 2,000,000 time-steps, or approximately 100 eddy turnover cycles, before sampling of data starts. Values are sampled every 40th time-step. Complete field data of standard deviations and mean values are exported every 200,000 time-steps.

5. Results

Here the results for the plane channel flow, porous media and the Rayleigh–Bénard convection are presented.

5.1 Plane channel flow

The validation case for the plane channel flow is carried out for $Re_\tau \approx 180$. A volume render of the instantaneous temperature field from the LBM simulations clarifies that the turbulence influences the temperature field near the wall, see Figure 8. Next the hydrodynamical statistics of the LBM case is validated against the DNS data presented in Kawamura *et al.* (1998). The hydrodynamical statistics of the LBM case (blue) and Kawamura *et al.* (dashed red) are presented in Figure 9. The components of the kinetic energy budget are the production (P), the dissipation (ϵ_k), the turbulent convection (T), the viscous diffusion (V) and the pressure transport (p_T) which are marked in the figure. For further details on the kinetic energy budget terms and their significance the reader is referred to Abe *et al.* (2001). Overall there is excellent agreement between the LBM simulations and Kawamura *et al.* data. Regarding U^+, which is defined in equation (38), the main difference is at small $y+$ values. Here, U^+ is larger for the LBM simulations because the LBM case near wall resolution is lower and the wall resolution is at a distance of $\approx y + = 1$. At the wall U^+ is, however, zero for both cases, which is, naturally, not captured with the logarithmic scale applied.

Regarding the Reynolds stresses $u_i' u_j'$ there are only minor differences for $u'2$ and $v'2$ for limited ranges of y^+. There are also minor differences for the energy budget terms. These smaller discrepancies are judged to be attributed to normal numerical errors. The comparison of the thermal statistics, see Figure 10, is likewise in excellent agreement with the same deviation as for U^+ at low y^+ due to the LBM wall resolution.

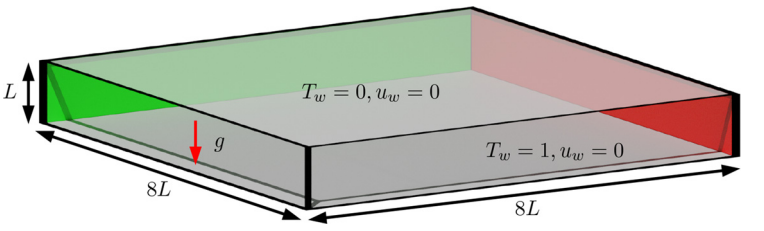


Figure 7.
Geometry and boundary conditions for Rayleigh–Bénard convection, red and green are periodic in x (red) and y (green) coordinate directions

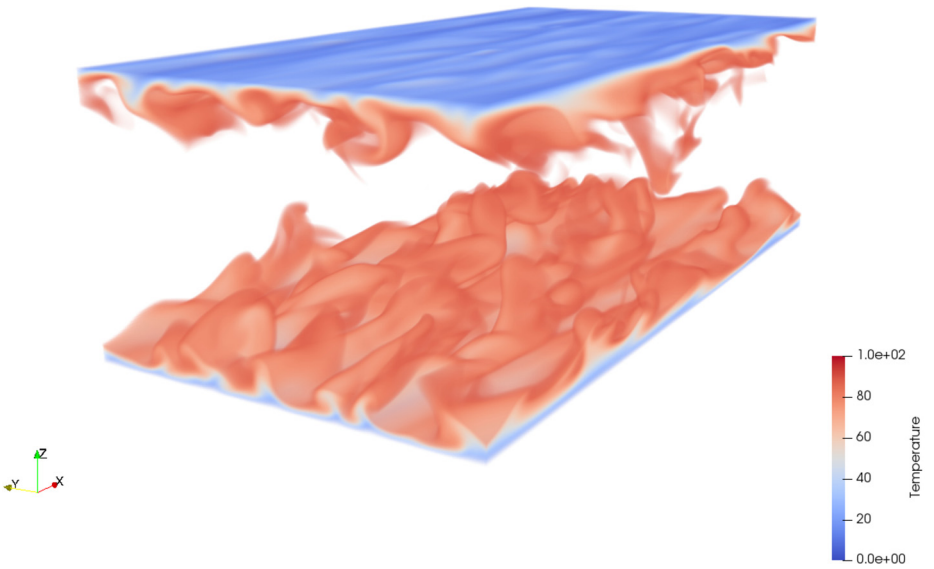


Figure 8.
Volume render of the final instantaneous temperature distribution for the plane channel case

Note: The walls have a temperature of 0°C (blue)

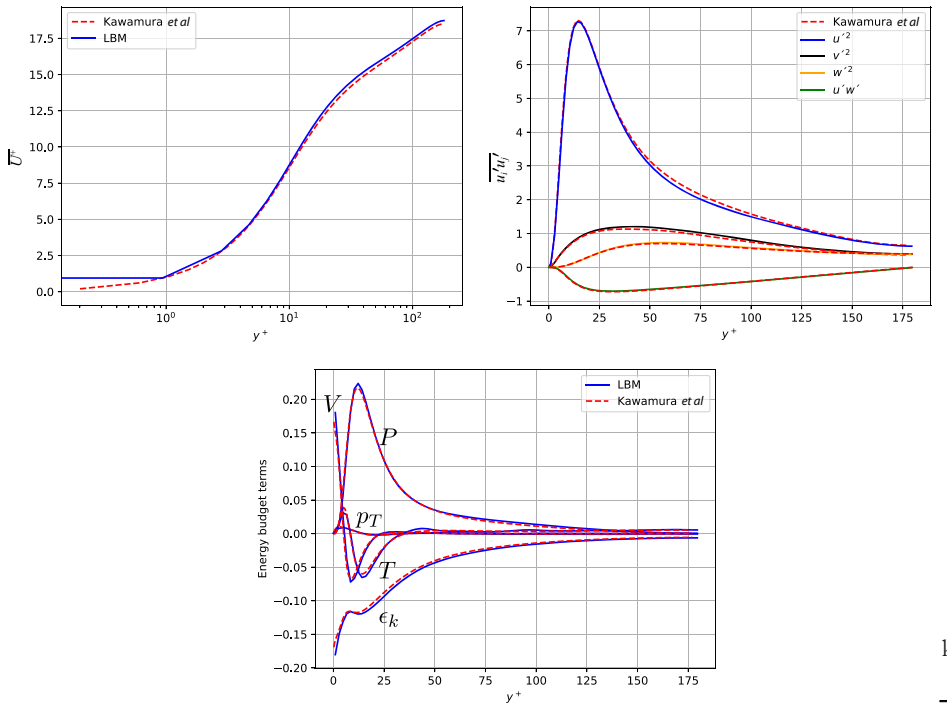


Figure 9.
Hydrodynamical statistics for plane channel flow, top left average velocity, top right Reynolds stresses and bottom kinetic energy budget terms

5.2 Porous media

For the porous medium studied there are complex interactions between the thermal field and flow field structures. This is particularly apparent in, for example, the horseshoe vortices folding across the cylinders, see the volume plot of an instantaneous thermal field in Figure 11. By averaging results from the LBM simulations, flow and thermal data over time can be compared to the pore scale data presented in Chu *et al.* (2019). This comparison yields

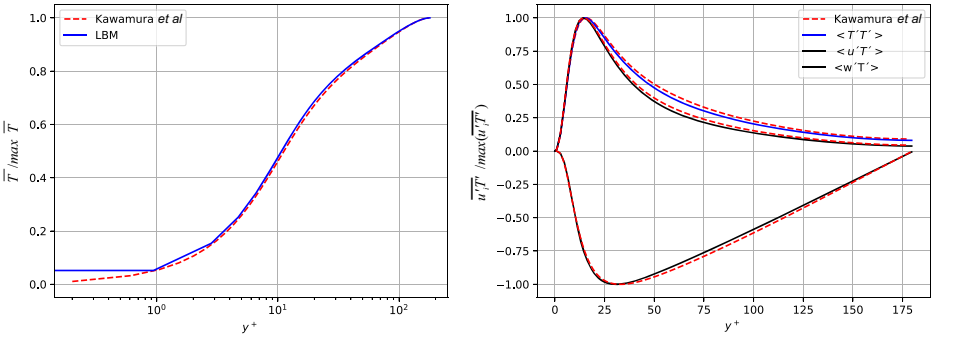


Figure 10.
Thermal statistics for plane channel flow, to the left dimensionless temperature and right temperature stresses

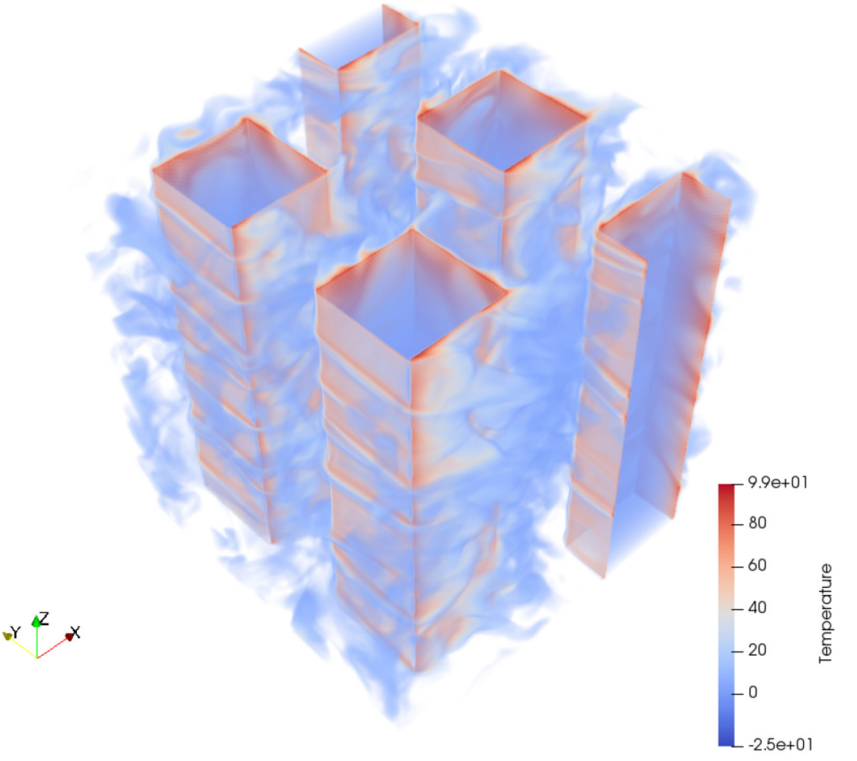


Figure 11.
Volume render of the instantaneous temperature distribution for the porous media case

that the average trends of the flows are the same, also in this case, indicating excellent agreement between the cases, see Figure 12.

5.3 Rayleigh–Bénard convection

An instantaneous thermal field for the LBM simulations of Rayleigh–Bénard convection can be seen in Figure 13. The temperature field takes the form of a typical Rayleigh–Bénard convection pattern consisting of plumes where the fluid sinks or rises, hence so-called Bénard cells are developed. A comparison to Wörner (1994) of the thermal statistics is presented in Figure 14. The comparison shows excellent agreement between the cases.

5.4 Performance and scaling

For GPU implementations, the goal is to have a maximum utilization of the memory bandwidth (Delbosc, 2015; Delbosc *et al.*, 2014). For this algorithm the theoretical maximum Lattice Updates per Second (LUPS) can be derived as:

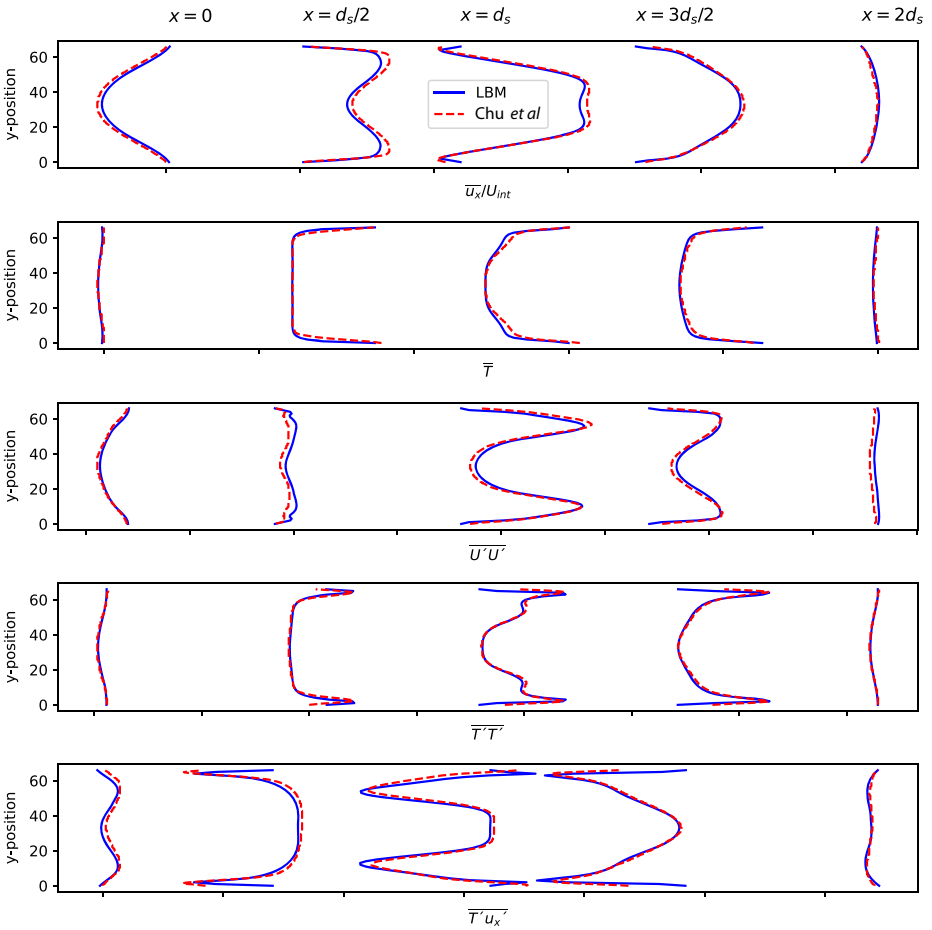


Figure 12.
Validation of the thermal and fluid dynamical pore scale statistics for porous media flow

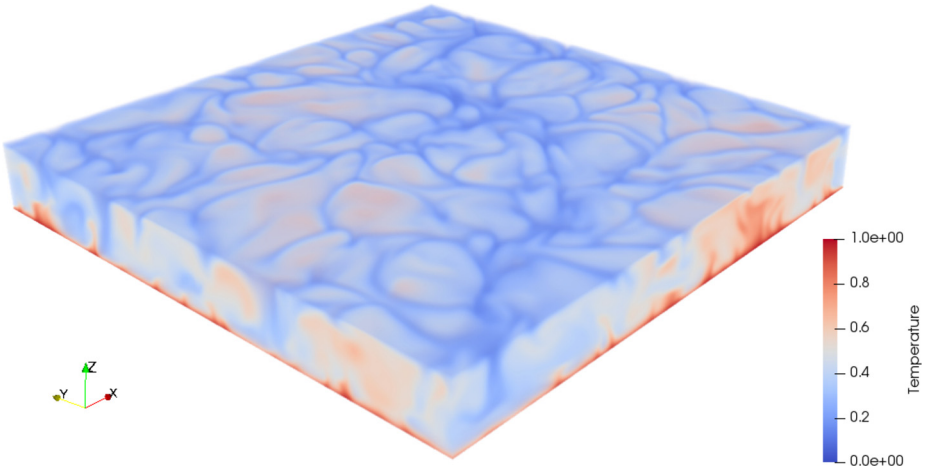


Figure 13.
A volume render of the instantaneous temperature distribution for the Rayleigh–Bénard validation case

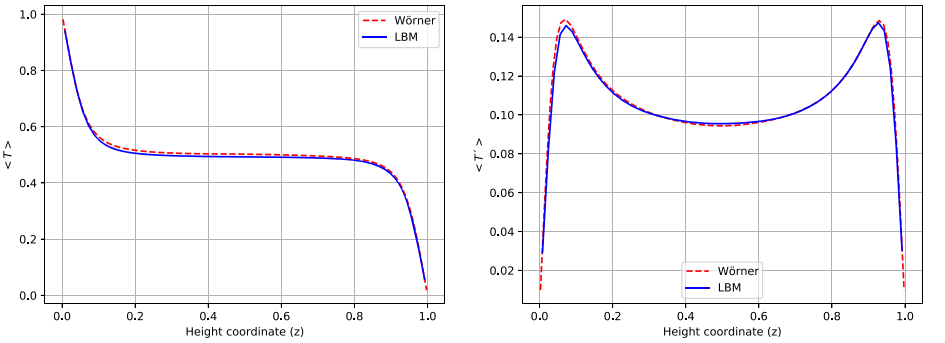


Figure 14.
Validation of thermal statistics for Rayleigh–Bénard convection at Ra = 630,000

Note: Average temperature as a function of height to the left and standard deviation of temperature as a function of height to the right

Table 2.
Performance in MLUPS for different cubic lattice sizes

n^3	*MLUPS*
16^3	490
32^3	1,417
48^3	1,710
64^3	1,831
128^3	1,979
256^3	1,996
384^3	1,977

$$LUPS = BW/(2f_n T_s). \tag{43}$$

where BW is the bandwidth of the graphics card, f_n is the amount of distributions being streamed and T_s is the size in bytes of the type being used for the distribution. For the RTX8000 cards used in this work, the theoretical maximum becomes $672e9/(2 \times 34 \times 4) \approx 2470MLUPS$. The performance is tested on a benchmark test case consisting of a cubic lattice of elements n^3, the case is initialized and then the code is executed for 2,000 time-steps. The longest execution timing, the $n^3 = 384^3$ case is approximately 60 s which means that thermal throttling effects are negligible. A summary of the values given from performance measurements can be seen in Table 2. The maximum performance is around 80% of the theoretical max capacity, this is the same utilization degree as presented in Delbosc *et al.* (2014) and Delbosc (2015). The performance begins to scale linearly when the amount of elements reaches approximately 1 M which is also on par with the results presented in Delbosc *et al.* (2014) and Delbosc (2015).

6. Conclusions

An MRT-LBM model with hydrodynamic (D3Q27) and thermal (D3Q7) lattices capable of simulating thermal turbulent flows has been implemented in the CUDA framework. The source code for the solver is openly available at the repository https://gitlab.com/c8383/thermal-mrt-lbm. The model is successfully validated with existing DNS data from three different turbulent thermal flow cases. Furthermore, the performance is on par with earlier published results from the area.

References

Abdul-Kadhim, A.A., Lien, F.S. and Yee, E. (2019), "GPGPU implementation of a lattice Boltzmann methodology for particle transport and deposition in complex flow", *International Journal of Numerical Methods for Heat and Fluid Flow*, Vol. 29 No. 7, pp. 2324-2351.

Abe, H., Kawamura, H. and Matsuo, Y. (2001), "Direct numerical simulation of a fully developed turbulent channel flow with respect to the Reynolds number dependence", *Journal of Fluids Engineering*, Vol. 123 No. 2, pp. 382-393.

Alcántara-Ávila, F., Hoyas, S. and Jezabel Pérez-Quiles, M. (2021), "Direct numerical simulation of thermal channel flow for Reτ = 5000 and Pr = 0.71", *Journal of Fluid Mechanics*, Vol. 916, pp. 1-22.

Bhatnagar, P.L., Gross, E.P. and Krook, M. (1954), "A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems", *Physical Review*, Vol. 94 No. 3, pp. 511-525.

Bouzidi, M., Firdaouss, M. and Lallemand, P. (2001), "Momentum transfer of a Boltzmann-lattice fluid with boundaries", *Physics of Fluids*, Vol. 13 No. 11, pp. 3452-3459.

Cerdá, J., Montoliu, C. and Colom, R.J. (2013), "LGEM: a lattice Boltzmann economic model for income distribution and tax regulation", *Mathematical and Computer Modelling*, Vol. 57 Nos 7/8, pp. 1648-1655.

Chen, S. and Doolen, G.D. (1998), "Lattice Boltzmann method for fluid flows", *Annual Review of Fluid Mechanics*, Vol. 30 No. 1, pp. 329-364.

Chu, X., Yang, G., Pandey, S. and Weigand, B. (2019), "Direct numerical simulation of convective heat transfer in porous media", *International Journal of Heat and Mass Transfer*, Vol. 133, pp. 11-20.

D'Humières, D., Ginzburg, I., Krafczyk, M., Lallemand, P. and Luo, L.S. (2002), "Multiple-relaxation-time lattice Boltzmann models in three dimensions", *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 360 No. 1792, pp. 437-451.

Delbosc, N. (2015), "Real-time simulation of indoor air flow using the Lattice Boltzmann method on graphics processing unit", Doctoral dissertation, University of Leeds.

Delbosc, N., Summers, J.L., Khan, A.I., Kapur, N. and Noakes, C.J. (2014), "Optimized implementation of the lattice Boltzmann method on a graphics processing unit towards real-time fluid simulation", *Computers and Mathematics with Applications*, Vol. 67 No. 2, pp. 462-475.

Fei, L., Du, J., Luo, K.H., Succi, S., Lauricella, M., Montessori, A. and Wang, Q. (2019), "Modeling realistic multiphase flows using a non-orthogonal multiple-relaxation-time lattice Boltzmann method", *Physics of Fluids*, Vol. 31 No. 4, p. 42105.

Forslund, T.O., Larsson, I.A., Hellström, J.G. and Lundström, T.S. (2021), "The effects of periodicity assumptions in porous media modelling", *Transport in Porous Media*, Vol. 137 No. 3, pp. 769-797.

Frisch, U., Hasslacher, B. and Pomeau, Y. (1986), "Lattice-gas automata for the Navier–Stokes equation", *Physical Review Letters*, Vol. 56 No. 14, pp. 1505-1508.

Gao, C., Xu, R.N. and Jiang, P.X. (2015), "Pore-scale numerical investigations of fluid flow in porous media using lattice Boltzmann method", *International Journal of Numerical Methods for Heat and Fluid Flow*, Vol. 25 No. 8, pp. 1957-1977.

Geier, M., Greiner, A. and Korvink, J.G. (2006), "Cascaded digital lattice Boltzmann automata for high Reynolds number flow", *Physical Review E*, Vol. 73 No. 6, pp. 1-10.

Ginzburg, I. and D'Humières, D. (2003), "Multireflection boundary conditions for lattice Boltzmann models", *Physical Review E*, Vol. 68 No. 6, p. 66614.

Hajabdollahi, F. and Premnath, K.N. (2018), "Central moments-based cascaded lattice Boltzmann method for thermal convective flows in three-dimensions", *International Journal of Heat and Mass Transfer*, Vol. 120, pp. 838-850.

Kawamura, H., Ohsaka, K., Abe, H. and Yamamoto, K. (1998), "DNS of turbulent heat transfer in channel flow with low to medium-high Prandtl number fluid", *International Journal of Heat and Fluid Flow*, Vol. 19 No. 5, pp. 482-491.

Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G. and Viggen, E.M. (2017), *The Lattice Boltzmann Method, Principles and Practice*, Springer International Publishing, Vol. 10 No. 978-3, pp. 4-15.

Li, Z., Yang, M. and Zhang, Y. (2016), "Lattice Boltzmann method simulation of 3-D natural convection with double MRT model", *International Journal of Heat and Mass Transfer*, Vol. 94, pp. 222-238.

Mohamad, A.A. (2011), *Lattice Boltzmann Method*, Springer, London, Vol. 70.

Premnath, K.N. and Banerjee, S. (2009), "Incorporating forcing terms in cascaded lattice Boltzmann approach by method of Central moments", *Physical Review E*, Vol. 80 No. 3.

Premnath, K.N. and Banerjee, S. (2011), "On the three-dimensional central moment lattice Boltzmann method", *Journal of Statistical Physics*, Vol. 143 No. 4, pp. 747-794.

Premnath, K.N., Pattison, M.J. and Banerjee, S. (2009), "Generalized lattice Boltzmann equation with forcing term for computation of wall-bounded turbulent flows", *Physical Review E*, Vol. 79 No. 2, p. 26703.

Qian, Y.H., D'Humières, D. and Lallemand, P. (1992), "Lattice BGK models for Navier–Stokes equation", *Europhysics Letters (EPL)*, Vol. 17 No. 6, pp. 479-484.

Suga, K., Kuwata, Y., Takashima, K. and Chikasue, R. (2015), "A D3Q27 multiple-relaxation-time lattice Boltzmann method for turbulent flows", *Computers and Mathematics with Applications*, Vol. 69 No. 6, pp. 518-529.

Tubbs, K.R. and Tsai, F.T. (2019), "MRT-lattice Boltzmann model for multilayer shallowwater flow", *Water (Switzerland)*, Vol. 11 No. 8, pp. 1-21.

Wörner, M. (1994), "Direkte simulation turbulenter Rayleigh-Benard-Konvektion in flüssigem Natrium", Karlsruhe 1994. (KfK. 5228.) Fak. f. Maschinenbau, Diss. v. 8.2.1994.

Yilmaz, I. (2020), "Parallel direct numerical simulation and analysis of turbulent Rayleigh-Bénard convection at moderate Rayleigh numbers using an efficient algorithm", *Computers and Fluids*, Vol. 213, p. 104754.

Yoshida, H. and Nagaoka, M. (2010), "Multiple-relaxation-time lattice Boltzmann model for the convection and anisotropic diffusion equation", *Journal of Computational Physics*, Vol. 229 No. 20, pp. 7774-7795.

Corresponding author
T.O.M. Forslund can be contacted at: tobfor@ltu.se