2013

LATTICE BOLTZMANN METHOD AND CELLULAR AUTOMATA SIMULATION OF PARTICLE MOTION AND DEPOSITION IN 2-D CASE

Yichao Deng
Michigan Technological University

LATTICE BOLTZMANN METHOD AND CELLULAR AUTOMATA SIMULATION OF PARTICLE MOTION AND DEPOSITION IN 2-D CASE

By
Yichao Deng

A REPORT
Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
In Mechanical Engineering

MICHIGAN TECHNOLOGICAL UNIVERSITY
2013

This report has been approved in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE in Mechanical Engineering.

Department of Mechanical Engineering-Engineering Mechanics

Report Advisor: *Dr. Song-Lin Yang*

Committee Member: *Dr. Kazuya Tajiri*

Committee Member: *Dr. Zhengfu Xu*

Department Chair: *Dr. William W. Predebon*

Acknowledgements

I would like to take this opportunity to thank all the people who were involved directly and indirectly in successful completion of my project. I am grateful to my advisor Prof. Song-Lin (Jason) Yang, whose expertise, guidance, patience and continuous encouragement helped me to achieve my target. He is always being an excellent teacher who developed a habit in me to think independently. Also he was an excellent reviewer of my work at every stage.

I would like to thank Prof. Zhengfu Xu and Prof. Kazuya Tajiri for being on my Master's defense committee.

I am also thankful to my wife Jinghong Zhang who takes care of my life and helps me with English part.

I sincerely thank my parents Huimin Deng and Jianlan Ji for supporting me and providing all sorts of help I need for successful completion of my Master's Degree. My brother Xue Wei is always being my source of inspiration. I would to thank my grandparents for their encouragement and moral support.

I am also thankful to all my friends in Michigan Tech and back home for their help and support.

Contents

Abstract

This technical report discusses the application of the Lattice Boltzmann Method (LBM) and Cellular Automata (CA) simulation in fluid flow and particle deposition.

The current work focuses on incompressible flow simulation passing cylinders, in which we incorporate the LBM D2Q9 and CA techniques to simulate the fluid flow and particle loading respectively. For the LBM part, the theories of boundary conditions are studied and verified using the Poiseuille flow test. For the CA part, several models regarding simulation of particles are explained. And a new Digital Differential Analyzer (DDA) algorithm is introduced to simulate particle motion in the Boolean model. The numerical results are compared with a previous probability velocity model by Masselot [Masselot 2000], which shows a satisfactory result.

Chapter 1

Introduction

The Lattice Boltzmann Method (LBM) is a new computational method used to simulate gases and fluids. The advantage of LBM primarily lies in its simplified treatment of complex geometry and locality for parallel programming, which greatly increases its efficiency in computation. The Lattice Boltzmann Method has evolved from Cellular Automata (CA), which can simulate complex phenomena simply by setting local rules for computational cells. The author intends to apply those two powerful techniques to simulate the particle deposition process. Furthermore, the simulation results can be used to simulate soot collection procedure in a Diesel Particulate Filter.

This report is divided into seven chapters. Chapter 2 gives a review of the theoretical study of LBM and CA. It includes the historical development and directions for future study of Lattice Boltzmann method and CA.

In Chapter 3, the author introduces the prevailing LBGK model and its theory foundation. Since this model is used in the later part of simulation, it is explained clearly in this chapter. The recovery of incompressible Navier-Stokes equation from LBGK model is derived in the form of finite difference which is too prolonged and attached in the appendix. Another two famous incompressible LB models are introduced in the appendix. Also the stability issue of LB methods is mentioned.

Boundary conditions are discussed in detail in Chapter 4. The feasible boundary conditions that can be applied in complex geometry are verified in Poiseuille flow test. Other famous boundary conditions not tested are explained in Appendix III.

Chapter 5 gives an introduction of particle deposition theory and numerical models. In this chapter, the particle motion loading the background velocity field generated by LB method is discussed. And the author introduces a new interpolation method for loading the flow field. The theory is established and illustrated with examples. Furthermore, a comparison is made between this model and the one

proposed by Masselot [Masselot 2000]. The author also gives an evaluation of this model and its future implementation.

In Chapter 6, we have the numerical simulation of single fiber deposition using the Boolean model.

We conclude the whole article in Chapter 7 and give a prospective of the proposed approach in the future.

Chapter 2

Literature Review

2.1 Lattice Boltzmann Method (LBM)

Since the 1970s the Boltzmann equation has been applied to kinetic theory through solving the distribution function to obtain the macroscopic flow field [Kanki & Iuchi 1973]. In order to solve Boltzmann equation, the full distribution function of each microscopic particle needs to be solved, which complicates the computation process. There are many other methods presenting the flow field from the physics of flow process instead of solving the Boltzmann Equation directly. Direct Simulation Monte-Carlo (DSMC) is the most famous among those methods [Bird 1994].

By ignoring the microscopic details in molecular dynamics, which has trivial influence in macroscopic dynamics, one simplifies the kinetic equation [Kadanoff 1986]. The Lattice Gas Automata (LG) has been developed since 1970s, which is considered as a simplified and fictitious molecular dynamic model [Hardy et al 1973, U. Frisch et al 1986]. McNamara & Zanetti [McNamara & Zanetti 1988] replace the particle occupation of Boolean variables with single-particle distribution functions of real variables and neglect individual particle motion and particle-particle correlations in the kinetic equations. This is the milestone which represents the birth of Lattice Boltzmann method.

Since then, more and more researchers have turned their attention to this new rising computational technique and perfected it in application. Through linearization of collision operator [Higuera & Jimenez 1989, Higuera et al 1989] and simplification of collision operator with single relaxation time [Bhatnagar et al 1954], the Lattice Bhatnagar-Gross-Krook (LBGK) model comes into being [Qian et al 1992]. In Qian's paper [Qian et al 1992], the authors state that the Navier-Stokes equation can be obtained at the second order of approximation with multi-scale technique.

He & Luo proposed a LBGK model for incompressible Navier-Stokes equation in 1997 [He & Luo 1997]. In this paper, the local pressure distribution function is introduced based on the equation of state (EOS). This model is widely accepted by people. In 2000, Guo described a new incompressible

model purely based on mathematics [Guo et al 2000]. The basic idea is similar to He & Luo's model but replaced with a new equilibrium distribution function.

The general theory and its extensions in applications are summarized by Chen [Chen & Doolen 1998] and Succi [Succi 2000]. These two articles are considered the important text in LBM area.

2.2 Cellular Automata (CA)

Wolfram [Wolfram 1983] interprets, Cellular automata are sufficient simple to allow detailed mathematical analysis, yet sufficient complex to exhibit a wide variety of complicated phenomena. Cellular automata shares the characteristics [Wolfram 1984]

- CA are regular arrangements of single cells of the same kind.
- Each cell holds a finite number of discrete states.
- The states are updated simultaneously (synchronously) at discrete time levels.
- The update rules are deterministic and uniform in space and time.
- The rules for the evolution of a cell depend only on a local neighborhood of cells around it.

Cellular Automata applies simple rules to local computational cells, which make this method quite efficient and easy to apply. Lattice Gas Cellular Automata is much more complex than Cellular Automata. General CA is usually too simple to simulate complex phenomena like fluids and gases. There is a need to apply some complicated rules to make CA satisfy the governing partial differential equations [Wolf-Gladrow 2005].

2.3 Diesel Particulate Filter (DPF)

Diesel Particulate Filter has come into being for almost three decades, aiming at reducing the particulate matter (PM) emission of the diesel engine. The most popular configuration of the DPF is the honeycomb wall-flow monolith which is shown in figure 2.1. The main deposition mechanisms in the DPF are those of Brownian diffusion and direct interception [Konstandopoulos 2000]. Konstandopoulos develop a 1-D model to simulate the deposition procedure in DPF. This model is tested in Catalyzed wall-flow diesel Particulate Filter (CPF) by Huynh et al [Huynh et al 2003] and the simulation results agree with the experimental data.

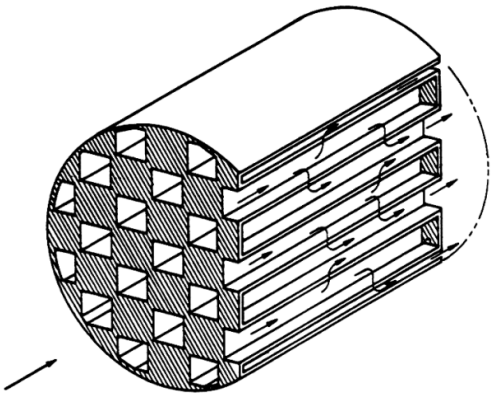


Fig.2.1 Honeycomb wall-flow monolith DPF (Konstandopoulos et al 2000)

Later on, the 3-D deposition model of DPF is simulated using LBM by Yamamoto et al [Yamamoto et al 2009]. And the authors also conduct extensive experiments to verify their model.

Chapter 3

Lattice Boltzmann Method

3.1 Lattice BGK (Bhatnagar-Gross-Krook) D2Q9 Model

D2Q9 stands for 2 dimensional 9 speeds model in LBM. Through discretization of velocity space, the velocity is restricted to finite directions. Figure 3.1 shows two dimensional lattice with discrete lattice velocities $\mathbf{e}_\alpha$ where $\alpha = 0$ to 8 represent 9 directions. Particle positions are confined to node positions. Hence variations in velocities in Cartesian coordinates for this model are given as:

$$\mathbf{e}_0 = \mathbf{0}, \quad \mathbf{e}_1 = c\mathbf{i}, \quad \mathbf{e}_2 = c\mathbf{j}, \quad \mathbf{e}_3 = -c\mathbf{i}, \quad \mathbf{e}_4 = -c\mathbf{j},$$

$$\mathbf{e}_5 = c(\mathbf{i}+\mathbf{j}), \quad \mathbf{e}_6 = c(-\mathbf{i}+\mathbf{j}), \quad \mathbf{e}_7 = c(-\mathbf{i}-\mathbf{j}), \quad \mathbf{e}_8 = c(\mathbf{i}-\mathbf{j}),$$

where c is the lattice speed $c = \delta_x / \delta_t$; δ_x and δ_t are the lattice constant in length and step size in time.

The single particle distribution function f_α along α direction can be thought to as a frequency of occurrence.

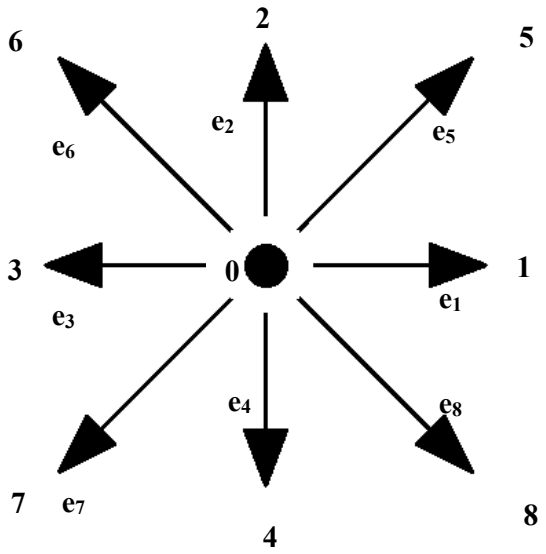


Fig.3.1 LBM D2Q9 lattice cell

In single relaxation BGK (Bhatnagar-Gross-Krook) approximation, particle distribution evolves due to collision tending towards an equilibrium distribution function which is defined by macroscopic velocity at that particular point. Collision contains only local information, satisfied mass, momentum and energy (etc.) conservation at each lattice site. Collision of fluid particles is considered as a relaxation towards a local equilibrium. The local equilibrium distribution function with a relaxation time determine the type of problem needed to be solved.

The D2Q9 (two dimensional nine speeds) lattice LBGK model use the equation of the system [He & Zou 1997]:

$$\underbrace{f_\alpha\left(\mathbf{x}+\mathbf{e}_\alpha\delta_t, t+\delta_t\right)}_{\text{Streaming}} = \underbrace{f_\alpha\left(\mathbf{x},t\right)-\frac{1}{\tau}\left[f_\alpha\left(\mathbf{x},t\right)-f_\alpha^{(eq)}\left(\rho,\mathbf{u}\right)\right]}_{\text{Collision}}$$

τ is the dimensionless relaxation time which is closely related to viscosity, and

$$\mathbf{e}_\alpha=\begin{cases}(0,0) & \alpha=0\\ \left(\cos[(\alpha-1)\pi/2],\sin[(\alpha-1)\pi/2]\right)c & \alpha=1,2,3,4\\ \left(\cos[(\alpha-5)\pi/2+\pi/4],\sin[(\alpha-5)\pi/2+\pi/4]\right)\sqrt{2}c & \alpha=5,6,7,8\end{cases}$$

are the particle velocity vectors.

The equilibrium distribution functions $f_\alpha^{(eq)}$ are

$$f_\alpha^{(eq)}\left(\rho,\mathbf{u}\right)=w_\alpha\rho\left[1+3\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c}\right)+\frac{9}{2}\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c^2}\right)^2-\frac{3}{2}\left(\frac{\mathbf{u}}{c}\right)^2\right],$$

where the weighting factors w_α are:

$$w_\alpha=\begin{cases}4/9 & \alpha=0\\ 1/9 & \alpha=1,2,3,4\\ 1/36 & \alpha=5,6,7,8\end{cases}$$

The density of each lattice unit and local momentum are given by

$$\rho=\sum_\alpha f_\alpha\ ;\qquad \rho\mathbf{u}=\sum_\alpha \mathbf{e}_\alpha f_\alpha\ ;$$

where $\mathbf{u}$ is the fluid velocity.

The evolution of LBGK model mainly consists of two steps: collision and streaming [Yu et al 2003]. Boundary conditions can be viewed as special collisions.

Collision: distribution functions f_α at the position $\mathbf{x}$ undergo the process by

$$\tilde{f}_\alpha\left(\mathbf{x}_i, t+\delta t\right)=\left(1-\frac{1}{\tau}\right)f_\alpha\left(\mathbf{x}_i, t\right)-\frac{1}{\tau}f_\alpha^{(eq)}\left(\mathbf{x}_i, t\right)$$

where $\tilde{f}_\alpha$ is the post-collision state, and $\mathbf{x}_i$ represents a point in the discretized physical space.

Streaming: post-collision distribution functions $\tilde{f}_\alpha$ move to the next position $\mathbf{x}+\mathbf{e}_\alpha\delta_t$.

$$f_\alpha\left(\mathbf{x}+e_\alpha\delta t, t+\delta t\right)=\tilde{f}_\alpha\left(\mathbf{x}_i, t+\delta t\right)$$

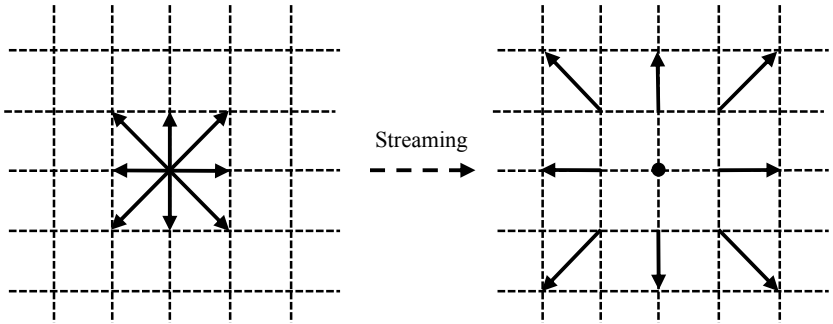


Fig.3.2 Streaming step [Sali 2012]

In principle, the relaxation factor 'τ' is a complicated functional of the distribution function f and $\tau=\lambda/\delta t$ ($\lambda\equiv$Relaxation time). Although streaming and collision can be combined into a single equation, they are strictly separated if solid boundaries boundary condition is applied. The BGK-type simple relaxation process in the kinetic equation allows the recovery of the non-linear convection in the Navier-Stokes equation through the multi-scale expansion.

The derivation of Navier-Stokes equation for the incompressible LBGK model using finite difference method is discussed in Appendix A. Since finite difference form is very clear while demonstrating the details of calculation, it can be applied in detailed analysis such as complex geometry boundary condition, pressure velocity condition and error analysis. Other governing equations can also be derived under different circumstance, such as time-independent and steady state conditions. One enlightening paper is written by He & Zou [He & Zou 1997] which shows that, under certain condition, the governing equations can be recovered through traditional finite difference methods.

The other two incompressible models proposed by He & Luo [He & Luo 1997] and Guo et al [Guo et al 2000] are introduced in Appendix II.

3.2 Stability and Convergence Issue

According to Yu et al [Yu et al 2003], the Courant-Friedrichs-Lewy (CFL) number is proportional to δ_t / δ_x, since the lattice units $\delta_t = \delta_x = 1$, which means the convergence speed of Lattice-Boltzmann Method is relatively slow. Some improvements have been made by Yu et al [Yu et al 2003], Bao [Bao & Schaefer 2008]. The common set-up of relaxation time in LBM is between one-half and two to reach a stable solution.

Chapter 4

Boundary Condition Disscusion

In this chapter we set up a Poiseuille channel flow test to examine prevailing boundary condition schemes. In numerical test part, we set up 100 lattice units in *x*-direction and 40 lattice units in *y*-direction.

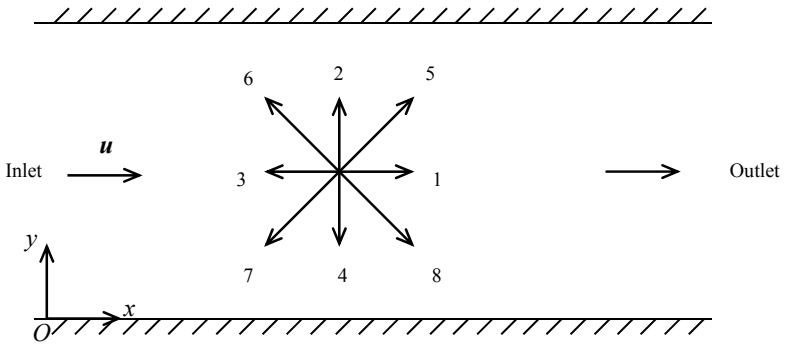


Fig.4.1 Schematics of Plane Poiseuille Flow

For a plane Poiseuille flow, the theoretical velocity distribution is given by:

$$u(y) = -\frac{1}{2\mu}\frac{dp}{dx}y(h-y).$$

where μ is the dynamic viscosity, h is the height of the channel, dp/dx is the pressure gradient; By setting $y = h/2$, we have the maximum velocity:

$$u_{\max} = -\frac{h^2}{8\mu}\frac{dp}{dx}.$$

To compare numerical solution with the analytical solution, we normalized the velocity distribution by their maximum velocity respectively.

4.1 Inlet and Outlet Boundary Condition

Zou & He proposed a generalized velocity pressure condition [Zou & He 1997]. The details of this boundary condition are introduced in following paragraphs.

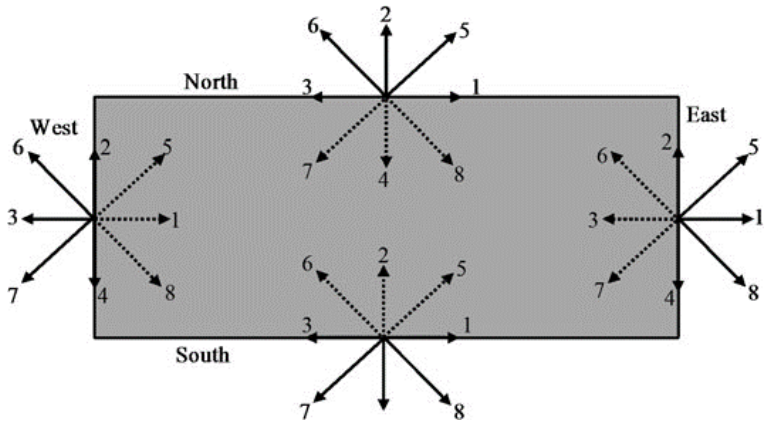


Fig.4.2 Boundary configuration for Poiseuille flow test [Mohamad 2011]

In the Poiseuille flow test, we set both the inlet (West) and outlet (East) constant pressure boundary condition. And therefore, according to Zou & He [Zou & He 1997], we have the equations below. Inlet (West): since f_2, f_6, f_3, f_7, f_4 and ρ_w are known after streaming, to derive f_5, f_1, f_8 and u_w, we have:

$$u_w = 1 - \frac{1}{\rho_w}\left[f_0 + f_2 + f_4 + 2\left(f_3 + f_6 + f_7\right)\right]$$

$$f_1 = f_3 + \frac{2}{3}\rho_w u_w$$

$$f_5 = f_7 - \frac{1}{2}\left(f_2 - f_4\right) + \frac{1}{6}\rho_w u_w + \frac{1}{2}\rho_w v_w$$

$$f_8 = f_6 + \frac{1}{2}\left(f_2 - f_4\right) + \frac{1}{6}\rho_w u_w - \frac{1}{2}\rho_w v_w$$

Similarly, we have the equations for the outlet (East):

$$u_e = 1 - \frac{1}{\rho_e}\left[f_0 + f_2 + f_4 + 2\left(f_1 + f_5 + f_8\right)\right]$$

$$f_3 = f_1 - \frac{2}{3}\rho_e u_e$$

$$f_7 = f_5 + \frac{1}{2}\left(f_2 - f_4\right) - \frac{1}{6}\rho_e u_e - \frac{1}{2}\rho_e v_e$$

$$f_6 = f_8 - \frac{1}{2}\left(f_2 - f_4\right) - \frac{1}{6}\rho_e u_e + \frac{1}{2}\rho_e v_e$$

4.2 Bounce Back Boundary Condition

There are several types of bounce back schemes for solid sites that can be applied to north and south walls. The most general bounce back can be viewed as on-grid bounce back scheme. We have an example regarding the bounce back scheme on North (Top) wall. If the solid nodes are assumed right

on wall, then we have the on-grid bounce back scheme.

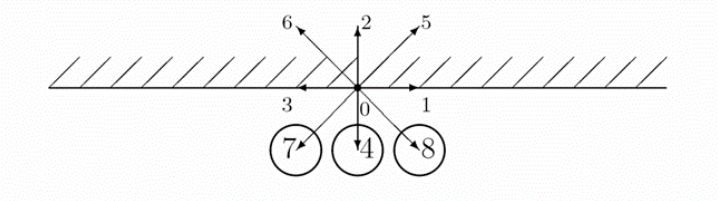


Fig.4.3 On-grid bounce back scheme [Sukop 2006]

In this case, we have the equation for on-grid bounce back [Succi 2001].

$$\begin{bmatrix} f_7(x,y) \\ f_4(x,y) \\ f_8(x,y) \end{bmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} f_5(x,y) \\ f_2(x,y) \\ f_6(x,y) \end{bmatrix}.$$

This bounce back scheme is based on the paper of Zou & He [Zou & He 1997], He et al [He et al 1997] and Chen & Doolen [Chen & Doolen 1998]. In their study, shifting position of solid wall by ½ lattice unit length will give second-order accuracy.

The result in Fig.4.4 is tested for channel flow described at the beginning of this chapter.

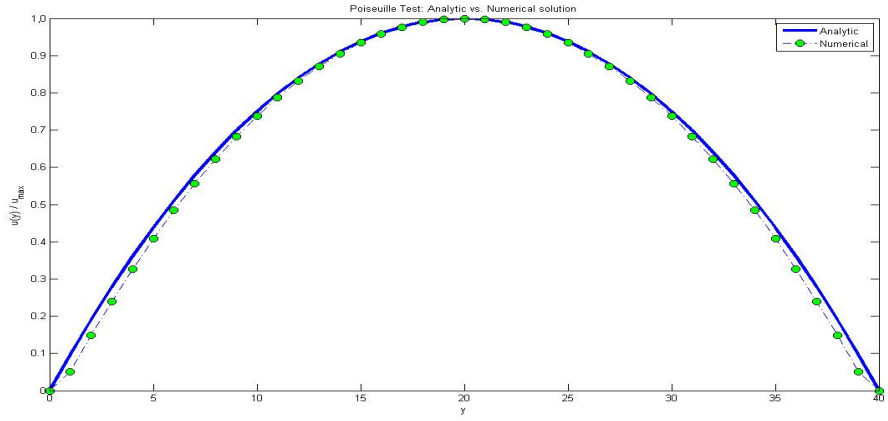


Fig.4.4 On grid bounce back scheme

The difference between analytical and LBM shown in figure 4.4 is called “slip velocity” [Inamuro et al 1995], and they proposed counter slip velocity scheme to eliminate the error. This scheme is introduced in Appendix III.

By setting 100 lattice units in *y*-direction, the result is shown in figure 4.5.

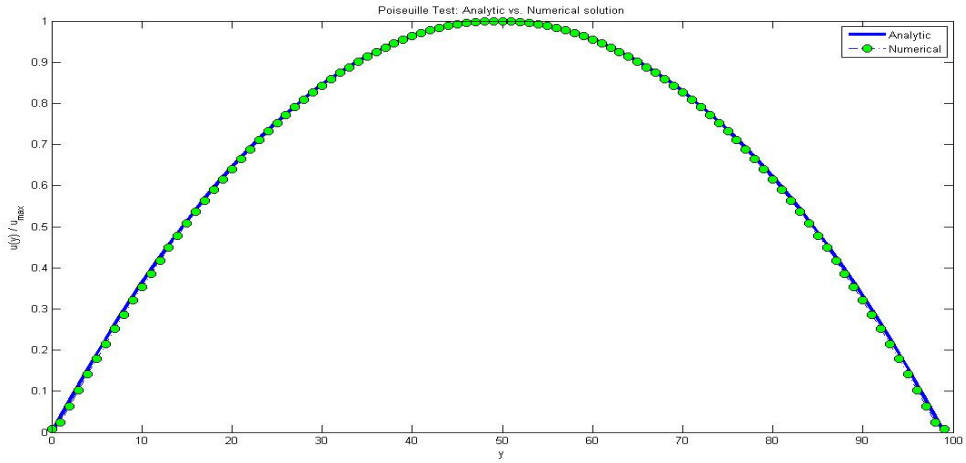
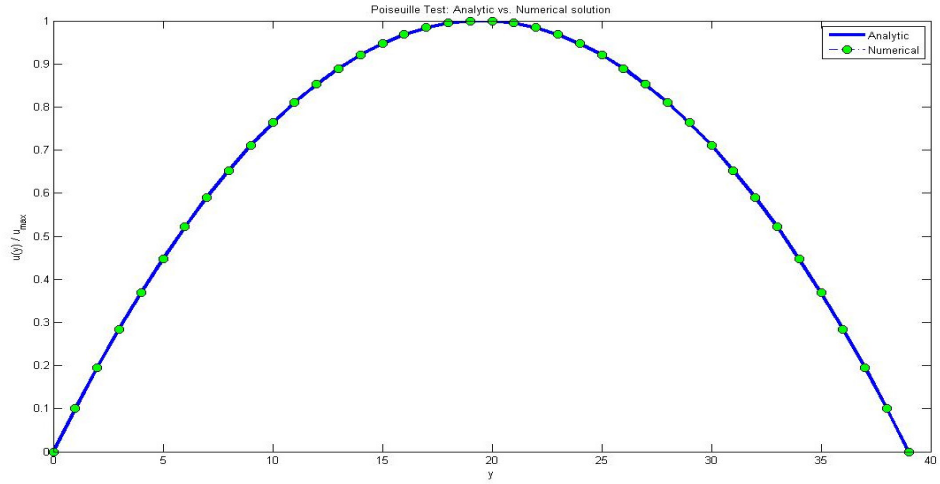


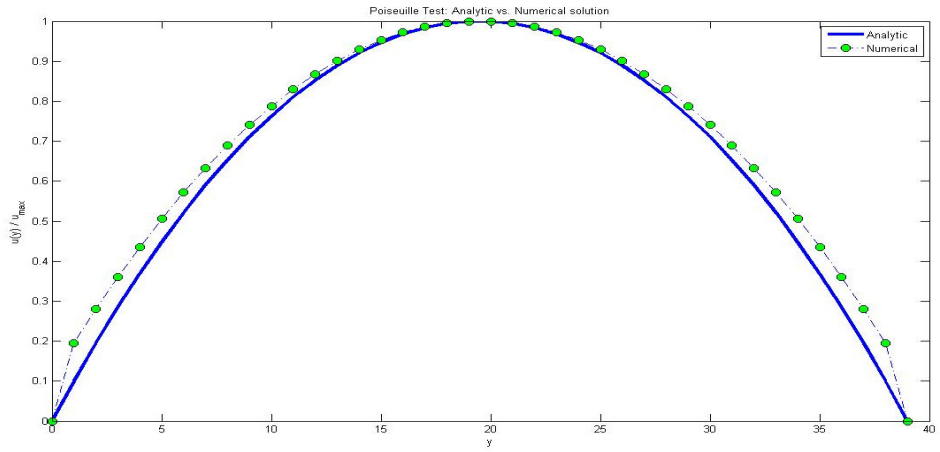
Fig.4.5 Result of using 100 lattice units in *y*-direction

The above result is consistent with the previous one using same bounce back scheme. By increasing the mesh size, the difference may be reduced.

He & Zou [He & Zou 1997] also examined the modified bounce-back scheme and showed it also has second-order accuracy. The difference of modified bounce-back scheme lies on the collision happens on the solid boundary nodes. And the results is only good for $\tau = 1$, For $\tau \neq 1$, there is a slip velocity.



(a)



(b)

Fig.4.6 Modified bounce-back scheme for (a) $\tau = 1$ and (b) $\tau = 0.6$.

Some other boundary conditions are discussed in the Appendix III.

4.3 Conclusion

From above discussion and testifying various schemes proposed by other researchers, due its simplicity, we will use the on-grid bounce-back scheme in this study.

Chapter 5

Particle Motion and Deposition

To model particle loading, two mechanisms are needed, namely particle motion and deposition.

To simulate the particle motion, the most common way is employing Lagrangian mechanics to calculate the motion of each individual particle directly. This method is very simple and easy to understand. However, once the number of particles engaged in is getting bigger, the collision part makes the complexity grows in the order $O(n!)$, which is impossible to calculate. However, once statistical methods are introduced, the Langevin equation can be integrated numerically [Ermak & Buckholz 1980]. And the complex statistical motion of the particles can be resolved.

The particle deposition mechanisms involve gravitational sedimentation, Brownian diffusion, inertial impaction, and direct interception [Friedlander 2000]. Brownian movement is essentially important for particles less than 1 micron [Ramarao et al 1994]. Some authors simulate dynamics of soot particles with the Langevin equation taking drag and stochastic forces of gas flow, gravity and thermo-phoretic force into account [Hayashi & Kubo 2008]. Others improved the numerical collection efficiency with relatively macroscopic experiments.

5.1 Boolean Model

The easiest way to model individual particles is to present single particle with single mesh unit which can perfectly fit in with the lattice system. However, the disadvantage is that the particle geometry and size cannot be modeled exactly. This scheme is derived from a Boolean model and its basic idea is similar to Cellular Automata (CA). Masselot [Masselot 2000] presented this model in his Ph.D. thesis.

5.2 Probability Density Function (PDF) Model

The particles are not treated individually, but as statistical occupation of space, no matter in volume or in mass. If the flow field is uniform, usually the probability of particle distribution is uniform, and we can employ a single value to represent how many particles are loaded in the flow field. This way of dealing with particle deposition greatly accelerate the computing, which explains why such schemes

are more easy-to-execute and less time consuming. Some authors use the concepts “bulk density”, “volume concentration” [Hayashi & Kubo 2008] and “soot concentration” [Yamamoto 2006]. Actually, such concepts behave instinctually the same.

5.3 Method of Moments (MoM) model

In the paper by Gschaider & Honeger [Gschaider & Honeger 2006], the authors take the change of particle size distribution into consideration. This model was originally proposed by Smoluchowski in 1917, which is the most efficient method in calculating the Brownian coagulation [Frenklach 2002]. The time-evolution of particle population was approximated by dividing the particles into infinite groups and described by a set of differential equations accordingly. Despite the accuracy and efficiency of MoM model, it is still very cumbersome for simulating the simple case. But this method may be applied in the future for improving the interactions between particles.

5.4 Masselot’s Model of Particle Motion

Since the velocity at each lattice point is given by Lattice Boltzmman method, the global velocity field can be obtained theoretically through traditional interpolation in CFD. Once the flow field is known, all kinds of particle deposition methods can be applied. Masselot proposed a way of dealing with the particle motion in his Ph.D. thesis [Masselot 2000].

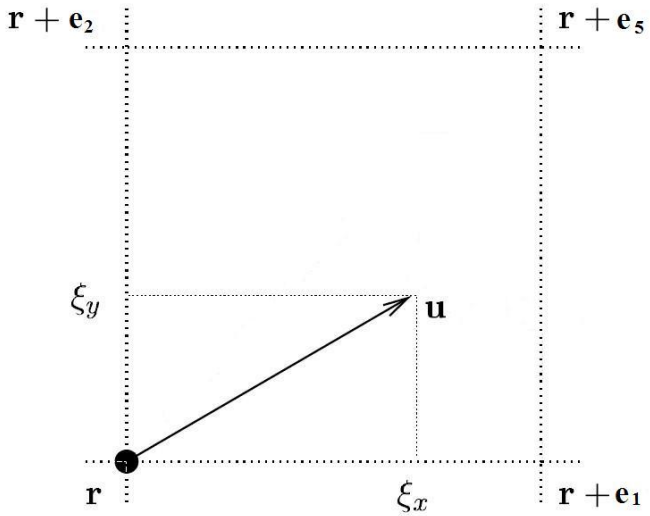


Fig.5.1 Particle motion probabilities related to velocity.

Masselot decomposes the velocity into x and y direction, represented by u_x and u_y respectively.

And also set up probabilities $\xi_x = |u_x|$ and $\xi_y = |u_y|$.

In this case, ξ_x represents the probability of which particle will move along x – direction, and respectively ξ_y represents the probability particle will move along y – direction.

The probabilities of resting or reaching other three neighbors in the quadrant are:

Probability moving to the diagonal site $\mathbf{r}+\mathbf{e}_5$: $p_5 = \xi_x\ \xi_y$

Probability moving to the x-neighbor site $\mathbf{r}+\mathbf{e}_1$: $p_1 = \xi_x\ (1-\xi_y)$

Probability moving to the y-neighbor site $\mathbf{r}+\mathbf{e}_2$: $p_2 = \xi_y\ (1-\xi_x)$

Probability remain at site $\mathbf{r}$: $p_0 = (1-\xi_x)\ (1-\xi_y)$

The author also provides justification of this method, estimates the predicted trajectory deviation from the path-line, and analytically approximates a mean diffusion coefficient based on the given algorithm.

Usually the velocity in Lattice Boltzmann method for the incompressible flow is always below 0.15 [He & Luo 1997] which means the propagation speed is relatively low. However, the efficiency can be increased by adding the time variable τ_s [Chopard & Masselot 1999, Chopard et al 2000]. The probability term p_i becomes $p_i(\tau_s / \tau)$, where τ is the time step in LBM, and the time variable τ_s is always greater than τ.

5.5 Digital Differential Analyzer (DDA) Algorithm Introduction

DDA (Digital Differential Analyzer) circular interpolation algorithm is often applied to NC (Numerical Control) machining trace control. In numerical control domain, Digital Differential Analyzer (DDA) algorithm is frequently used for linear interpolation of variable over an interval between start and end point. In traditional CNC machine step, motor is the actuator, which means the movement can only be transferred as binary signals, the interpolation process should be interpreted with Boolean operation of the trajectory path [Suh et al 2008].

Regardless of the interaction between fluids and particles, for each individual particle, the path-line is known. Instead of DDA algorithm, of which the trajectory path behaves like path-line, we may use the register concept to realize the velocity at each lattice point. Data registers can hold numeric values

such as integer and floating-point values and other data. In some older and low end CPUs, a special data register, known as the accumulator which could keep adding numbers. It will clear itself when the register is full.

If applying Lagrangian approach to each particle, it is necessary to set a fictitious "register" for every particle that enters the calculation domain. The number of "registers" increases as the more particles entered, which makes re-allocation cumbersome for the computer RAM. Also the trajectories of the particles need to be tracked in this algorithm. All these disadvantages make this description hard to realize.

If we apply the Eulerian specification, the number of registers needed is coupled with the mesh size which is fixed. And the particle trajectory does not need to be tracked. This approach seems to facilitate the calculation. But we need to verify this approach in the later part.

We have an example for DDA interpolation of particle motion in a 2-D uniform velocity flow field, the *x*-direction velocity component is $u_x = 0.12$ and *y*-direction velocity component is $u_y = 0.05$. For the two dimension case, we need to set up two registers for *x*-direction and *y*-direction. The registers need to set a threshold value; usually this value is no less than the maximum velocity in the flow field, and we set it $u_{\max} = \sqrt{u_x^2 + u_y^2} = \sqrt{0.12^2 + 0.05^2} = 0.13$. Also the register needs to be clear to 0.

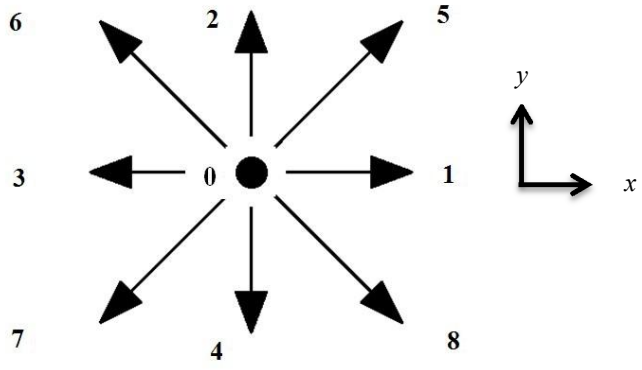


Fig.5.2 Schematic of particle motion

In above figure, a particle enters the site "0", and the velocity at this point is $u_x =$ 0.12 and $u_y =$ 0.05. For this particle, it can only move to sites "1", "2", "5" or remain at "0".

Step (1): The registers begin to work,

add *x*-direction velocity to *x*-direction register

ddax = 0 + 0.12 = 0.12;

add *y*-direction velocity to *y*-direction register

dday = 0 + 0.05 = 0.05.

Step (2): Then judge whether any register is full,

ddax (0.12) < *u_max* (0.13)

dday (0.05) < *u_max* (0.13).

Step (3): This particle will remain at "0" site.

Again, since the particle is still at this site, the registers at site "0" keep working. Repeat above steps:

Step (1):

ddax = 0.12 + 0.12 = 0.24;

dday = 0.05 + 0.05 = 0.10;

Step (2):

ddax (0.24) > *u_max* (0.13)

dday (0.10) < *u_max* (0.13)

Step (3): Since *ddax* > *u_max* and *dday* < *u_max*, which indicates only the *x*-direction register is full. this particle will move along *x*-direction to site "1" and the register *ddax* should minus the threshold value *u_max*, 0.24 – 0.13 = 0.11.

Repeat above steps, we have the following positions of particles: (0, 0), (1, 0), (2, 1), (3, 1), (4, 1), (5, 2)…

And we plot them in a single figure which gives:

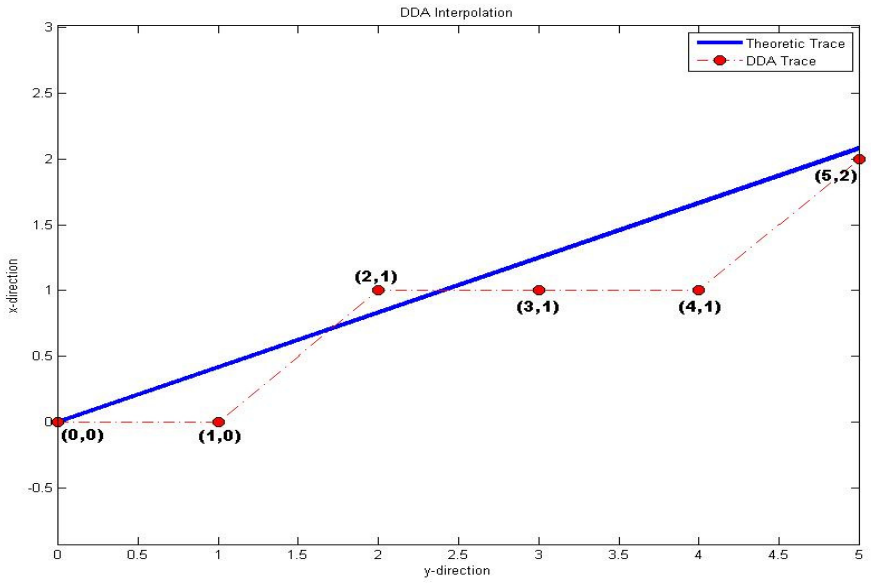


Fig.5.3 DDA interpolation trace

The theoretic trace is the path-line for particle at position (0, 0):

$$y = \frac{5}{12}x .$$

The program of the DDA method is attached in Appendix IV. And the flow chart for the program is given in Fig.5.5.

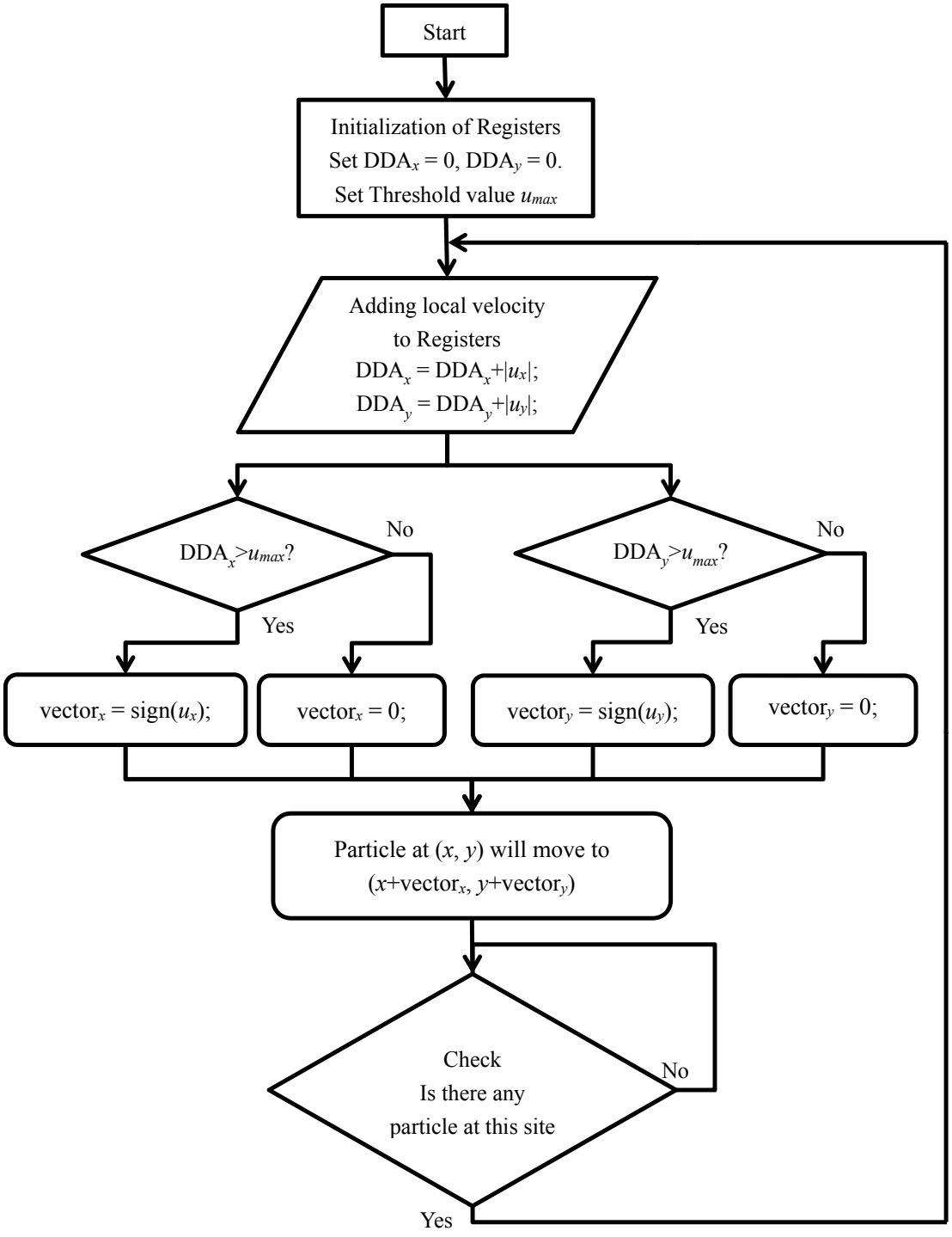


Fig.5.4 Flow chart of DDA algorithm

5.6 Modified Probability Motion

Based on the Masselot's probability motion model and the particle movement logic of DDA algorithm, we implement this hybrid scheme and call it the Modified Probability Motion model. The program of

this model is attached in Appendix IV. The flow chart is shown in Fig.5.6. This scheme is tested in the following part.

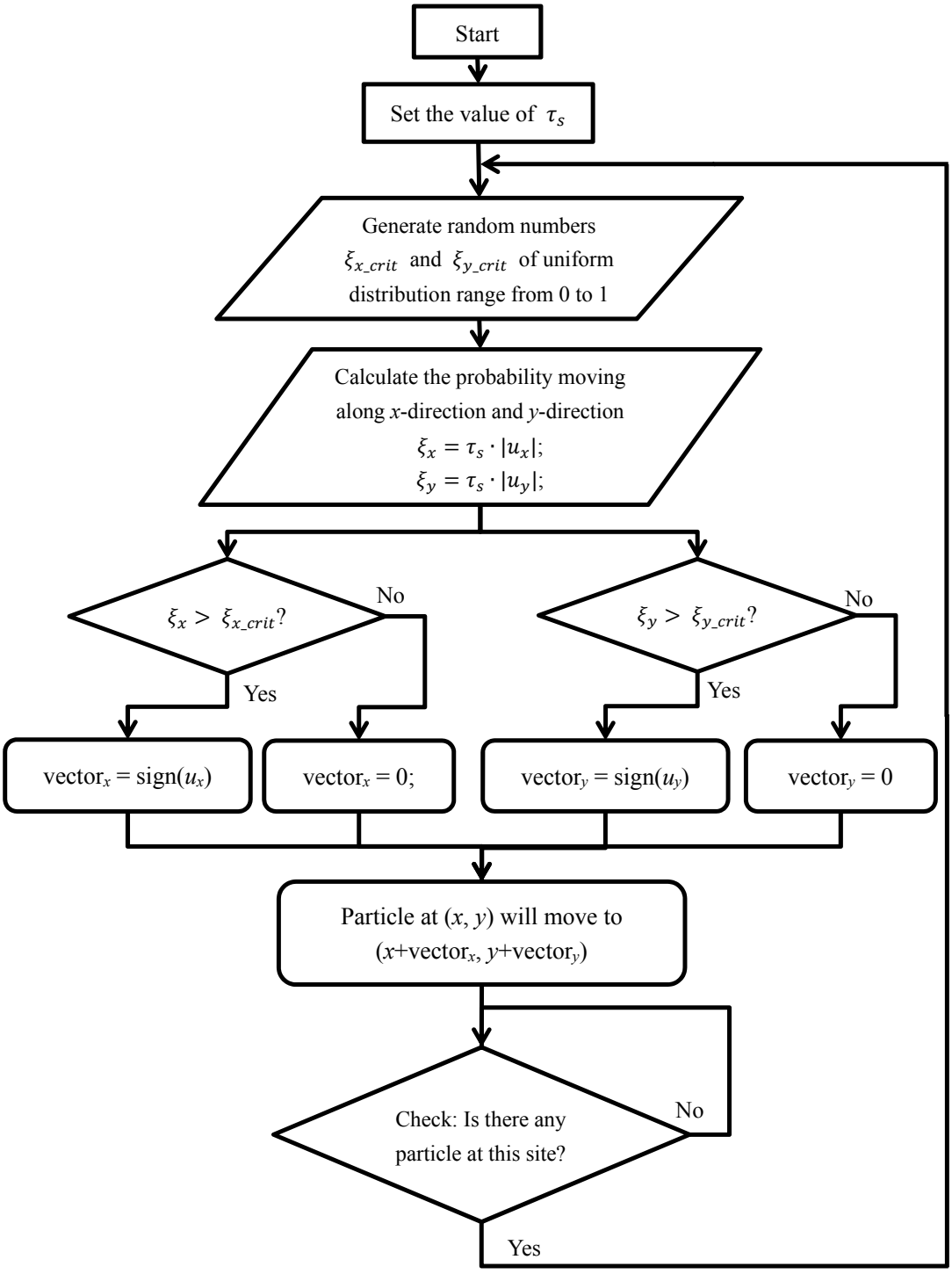


Fig.5.5 Modified Probability Motion flow chart

We implemented both DDA and Modified Probability Motion schemes; and tested them for an incompressible flow over a cylinder. The calculation domain consists of 300*150 lattice cells. The following figures show that, for single particle loaded in the flow field, what the motion trace would be like is contrary to the streamline.

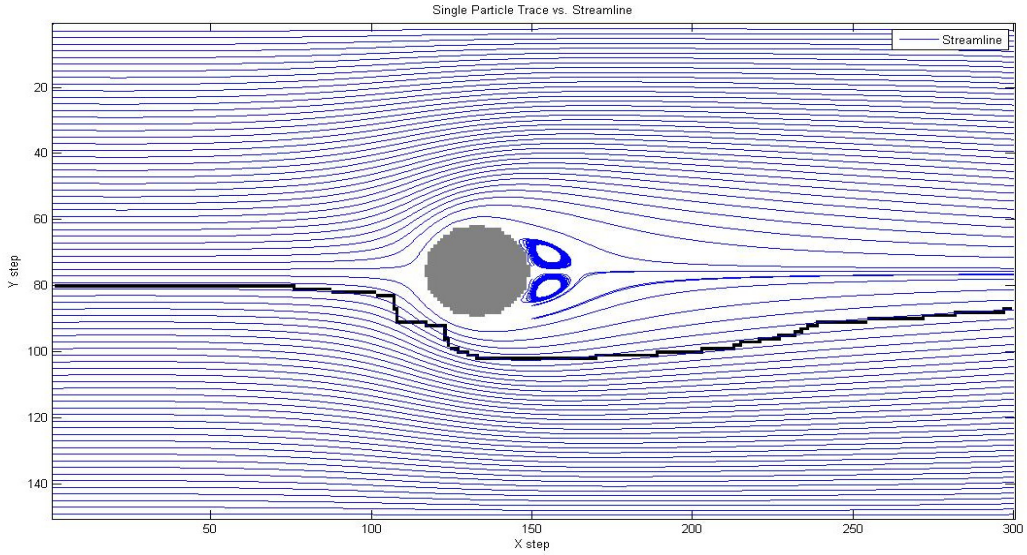


Fig.5.6 Single particle trace calculated by Modified Probability motion.

The gray circle is the solid part. The black solid line is the trace of the single particle we tracked in the flow field.

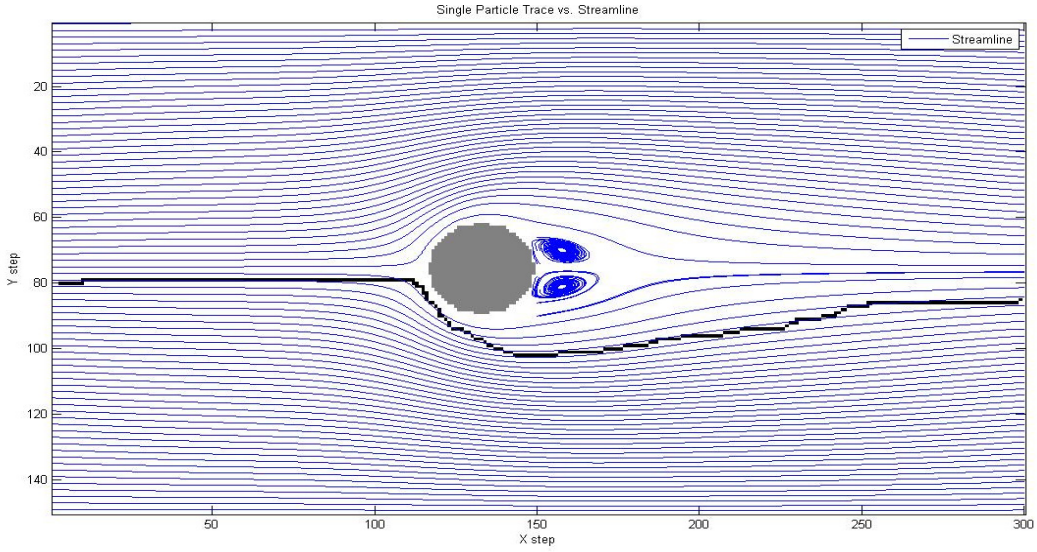


Fig.5.7 Single particle trace calculated by DDA method.

From above results, due to the nature of DDA scheme, as expected, the DDA method seems to follow the streamline more faithfully than the modified probability motion model. Theoretically, the DDA method is a relatively deterministic approach. However, in real flow with many particles, Brownian diffusion, which can be modeled as a random process due to inter-particle collision, will always exist and needs to be included in any models.

The author also noticed that the DDA method might have a little “inertial” effect due to the registers and Eularian configuration of the registers. The register behaves as an approximation of integration. When the velocity field changes, the register has immediate response. But the particle movement may be delayed because of the threshold value. So the response of particle movement is relative slow in terms of the velocity field. This kind of feature is not totally a defect. For simulating large mass particles, which indicates the inertial force begins to dominate the deposition procedure; this characteristic of inertial could be improved to fit the new case. This is only a future prospect; inertial effects are not discussed in this article.

Another issue is concerned with the initialization of the register. In CNC machine the initialization of the hardware register has three modes: empty, half-full and full. But in software interpolation, we have the privilege to set it a random number, which will greatly increase the accuracy and stability of the interpolation algorithm. We also test the half-full initialization, and the result shows there might be several waves resembling the mechanical vibration. However, after employing the random initialization, the vibration effects can be eliminated. Therefore, the random initialization may be the best way we can choose.

5.7 Deposition Scheme Discussion

In terms of combining the deposition process with the flow field obtained by Lattice Boltzmann methods, there are basically two ways of dealing with particle deposition regarding the lattice issues. One is to calculate each individual particle including all parameters such as particle geometry, rotatory inertia, collisions between particles. Once we have the velocity field, no matter which scheme was used, we can obtain the velocity at each random location using traditional stagger scheme. Then load

particle parcel data including size, density, geometry, location at the inlet. Since it is impossible to calculate each gas particle, we may employ Monte-Carlo method generating random fluctuation to simulate the Brownian motions. This kind of scheme is similar to the direct numerical simulation. Though DNS is very accurate according to the basic physical laws, it is quite costly, sometimes not even applicable with respect to computation resources. Therefore, more and more algorithms need to be developed to facilitate or to simplify the mechanism.

Another way of coping with the particle deposition is to use a simplified version of above scheme. Particles are considered only in on-lattice condition, no matter for the Boolean model or for the probability density model. Once particles reach the surface, or neighboring sites of the solid sites, they would be trapped and the deposition area begins to accumulate. For the Boolean model, since particles are represented in the value "1", the particles would immediately become solid sites once they enter the deposition area. In Masselot's paper [Masselot 2000], he sets a threshold value, say *N*. Only after *N* particles deposited would the site become solid. This way of handling deposition process is similar to the probability density function method. The value of probability density function is always between 0 and 1, the accumulation process is the same as stated above. And once the value reaches unity, the site will become solid. This method is also applied by Yamamoto [Yamamoto 2006].

The definition of neighboring sites may vary. And it may also influence the deposition pattern greatly. Referring to Fig.5.3, for the solid site "0", usually the sites "1", "2", "3" and "4" are sure to collect particles, since they are the closest neighboring sites. But the diagonal sites "5", "6", "7" and "8" may depend on what kind of deposition pattern we want. These two ways are tested in the programs, and we finally choose all the sites as collecting area.

5.8 Brownian Motion

The Brownian motion in the Boolean model is generated by random numbers which indicate where the particles should move. Since the lattice unit is the smallest length particle that may move, the Brownian diffusion rates may be controlled by distribution of random numbers generated.

For the Probability Density model, the Brownian diffusion for the particles may be easier to fulfill. In

Yamamoto's paper [Yamamoto 2006], the authors also apply the Lattice Boltzmann equation to the mass fraction, which indicate the Brownian diffusion is the same as that in LBM methods. This approach is similar to the LBM method applied in multiphase modeling, which decouple the reaction and flow parts. We also use the weighting coefficients as the diffusion rate coefficients.

Chapter 6

Numerical Simulation

The tested calculation domains are imported by MATLAB by bitmap files. And those bitmap files can manually drew in Visual Studio and Windows Painting. Though the unit 8 system uses the value 0 to present the color black which is inconvenient for us to use in programming, we may convert those values through logical negation operation. And we have the solid site of value "1" and flow domain of value "0".

6.1 Flow Passing a Cylinder

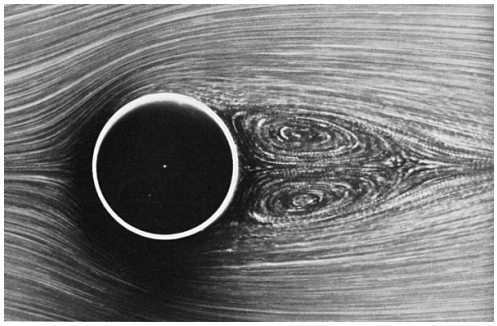
Firstly, we use the planer flow passing a cylinder as a test domain. The domain includes 300*150 pixels which are according to 300*150 lattice units.

For the LBM part, the relaxation time is set to be 1. The inlet boundary condition is set to be constant velocity $u_{inlet} = 0.1$. The upper and lower boundaries are applied with periodic boundary condition. The approximate diameter of the cylinder is $D = 30$. The kinematic viscosity is $\nu = \delta_x^2 / 6\delta_t = 1/6$. Therefore, the Reynolds number is given by

$$\mathrm{Re} = \frac{u_{inlet} D}{\nu} = 18.$$

The unit system of lattice Boltzmann method obeys dynamic similarity. Therefore, there is no need to worry about the units. The idea is expressed in the books by Succi [Succi 2000] and Sukop [Sukop 2006].

The velocity and streamline are presented below in compare with experimental result:



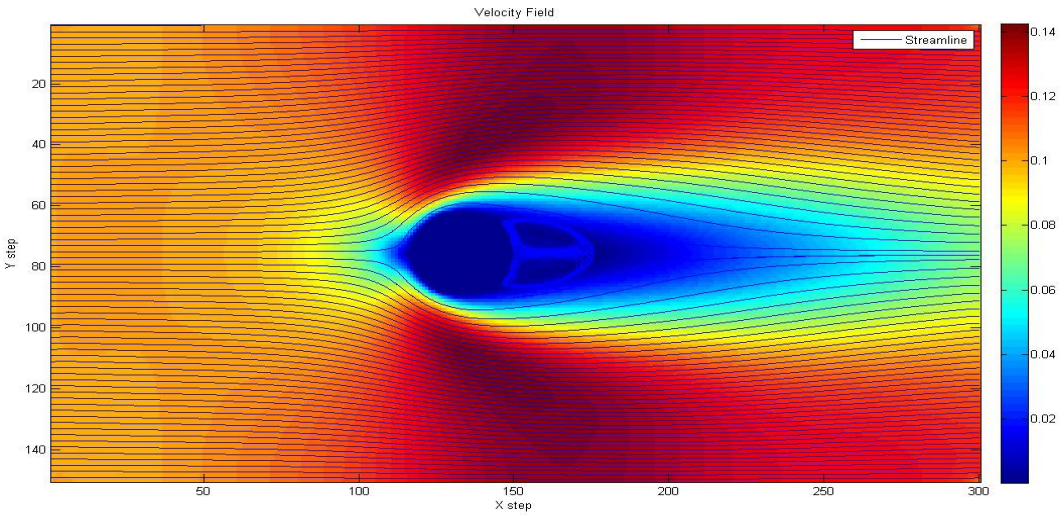
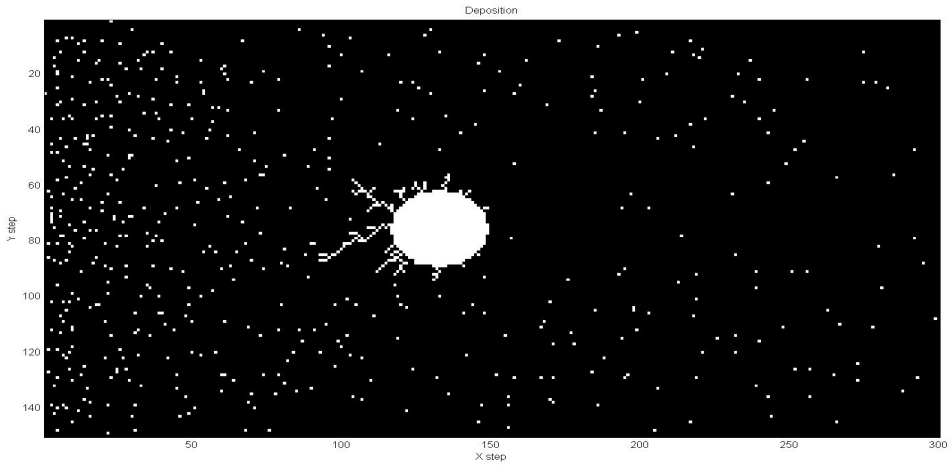


Fig.6.1 flow past a cylinder

Once the particles are introduced at the inlet of the flow field, they would move along with the fluid.



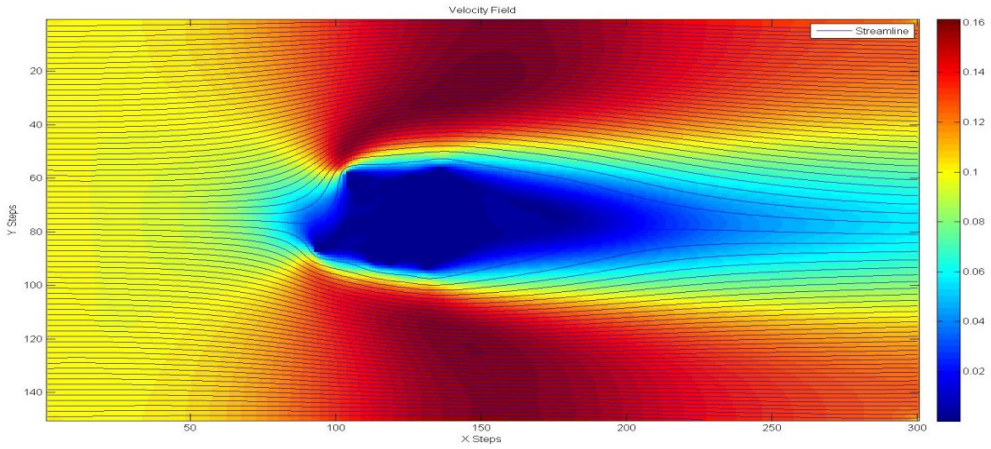


Fig.6.2 This figure shows the particle deposition using the binary DDA model

and the lower one shows the according velocity field with streamlines.

Figure 6.2 shows the particle deposition by the Boolean model we mentioned in the previous chapter. The white part represents the particles and solid part. And the black area is the flow domain. From above figures we may concern about the deposit pattern generated by the Boolean model. The results are similar to the results shown in Przekop's paper [Przekop et al 2003]. The dendrite deposit pattern can also be controlled by different diffusion rate and propagation speed of particles.

However, the deposit structure is a little weird since the solid is too obtrusive. We consider that the shear stress and impaction may also influence the deposit. Recall the shear stress:

$$\tau_w = \mu\left(\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}\right).$$

The stress is proportional to the velocity gradient. In LBM method, the shear stress near the wall is proportional to the velocity of the neighboring sites in fluid. Therefore, we may use the neighboring velocity as a criterion to judge whether the particles deposited would be blown away. And we have

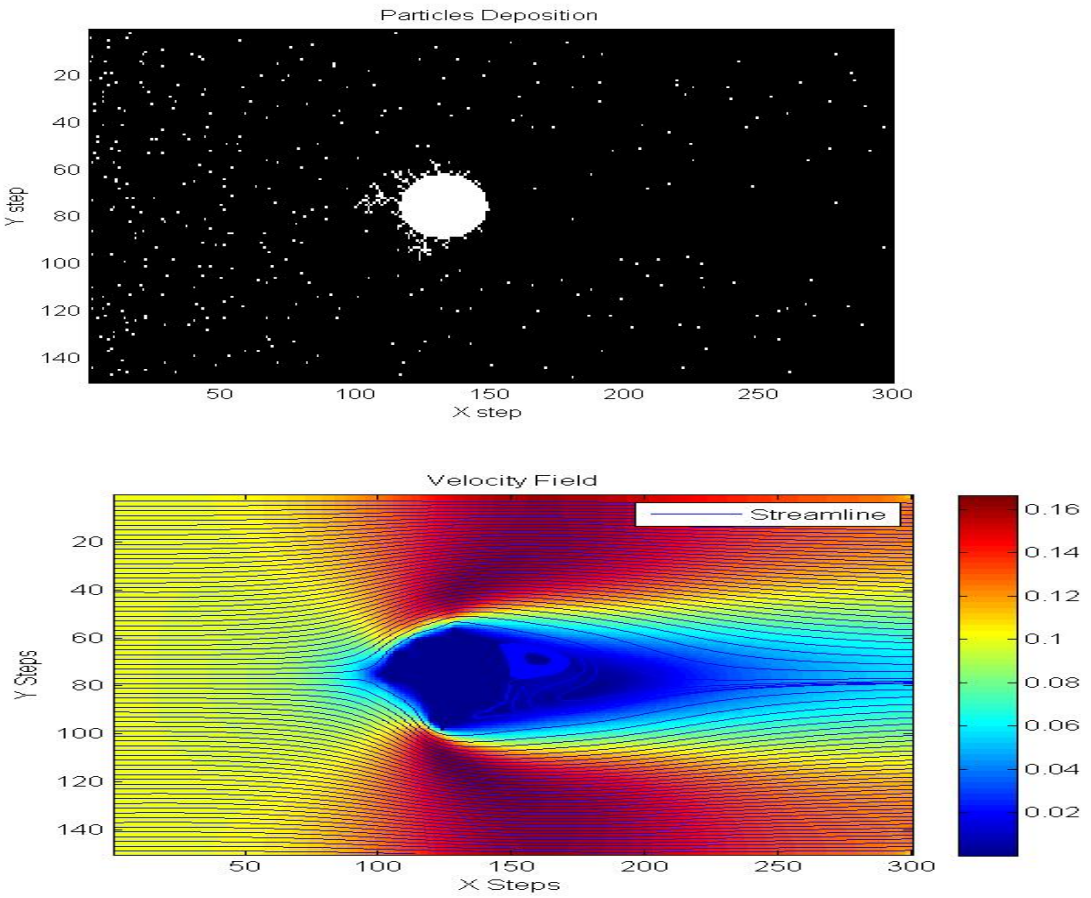


Fig.6.3 The deposition pattern after introducing the shear stress

From Fig.6.3 we may find the deposition pattern is modified a little. However, the 2-D Boolean model may not present the actual deposit pattern for single fiber. Applying CA rules to deposition procedure might facilitate computation of complex phenomena, but not guarantee the correctness. The deposition rules need to be testified.

For the Modified Probability Motion model, the deposition coupled with artificial Brownian diffusion is shown in Fig.6.4.

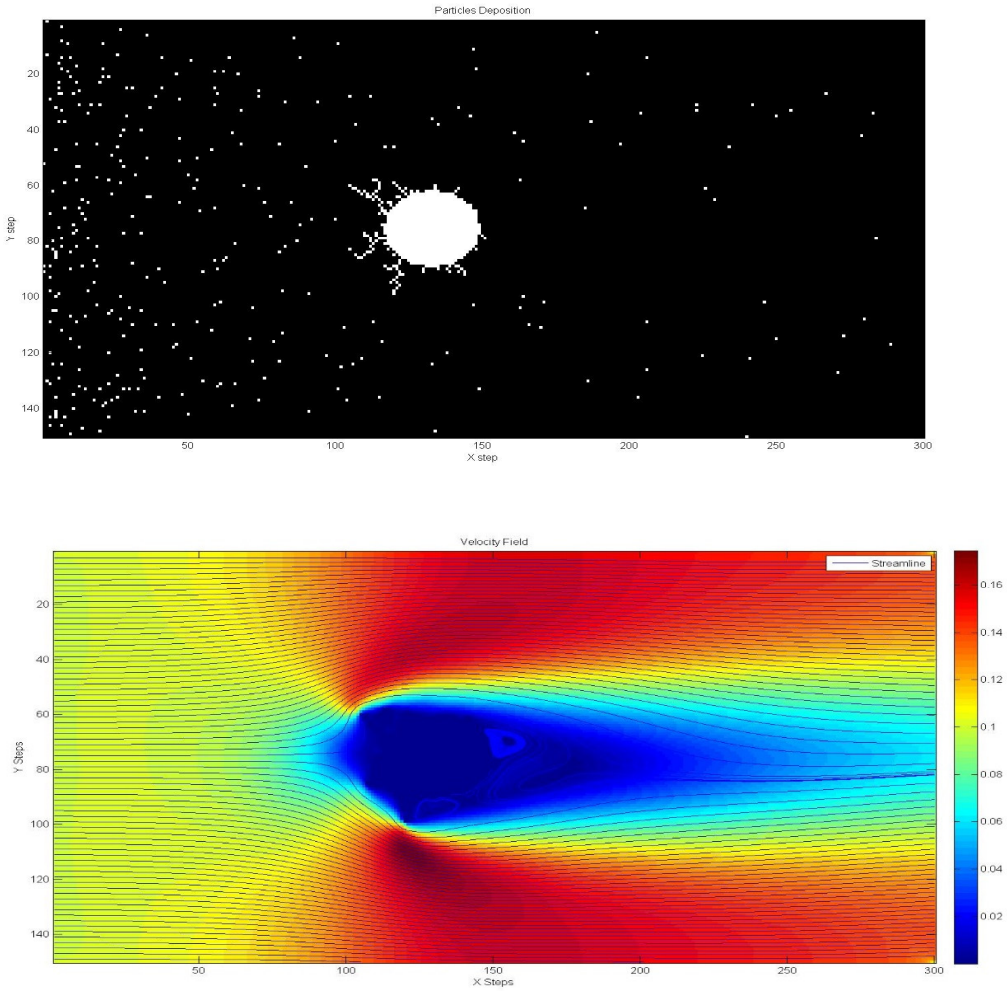


Fig.6.4 The particle deposition using Modified Probability motion with no artificial Brownian motion

Fig.6.4 shows similar results in compare with Fig.6.2. The dendrite pattern may change greatly due to the random number generated.

6.2 Porous Media Test

For the porous media problem, the artificial porous media is also generated in a bitmap file by Sali [Sali 2012]. The file consists of 150*150 pixels, where the black area represent the solid fibers. The porosity of this fibrous media is 0.7754.

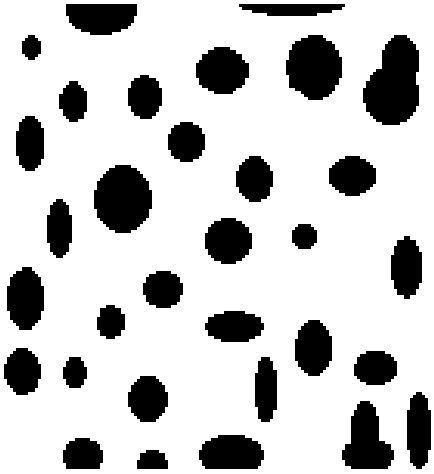


Fig.6.5 Artificial porous media for test [Sali 2012]

We set the inlet velocity to be $u_{inlet} = 0.01$. And the outlet is maintained at constant pressure $\rho = 1.0$. The upper and lower boundary has the periodic boundary condition.

The velocity field of the porous media is given by:

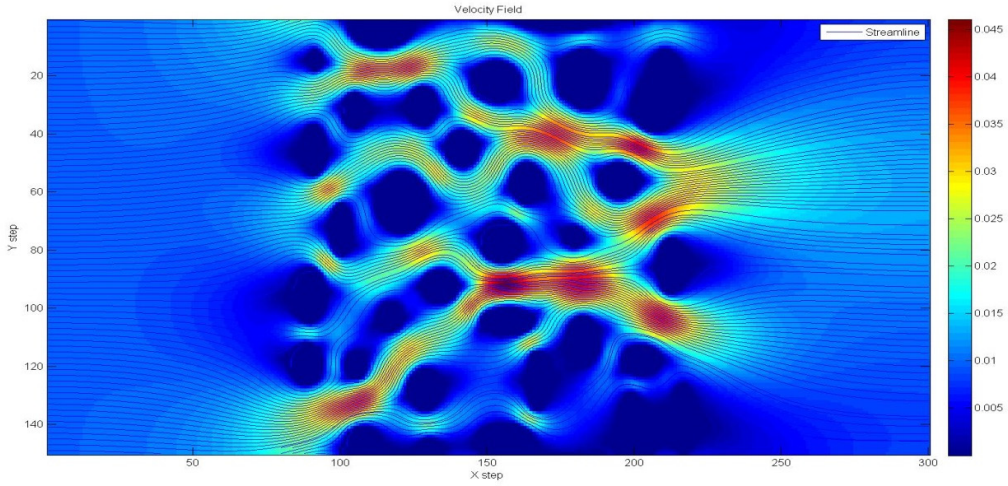


Fig.6.6 Velocity field of the porous media

Similar as the deposition procedure in previous section, we load the particles in the flow field. Firstly, we load the particle using DDA method with no artificial Brownian motion or shear stress peeling-off. The velocity field and deposition pattern are shown in Fig.6.7.

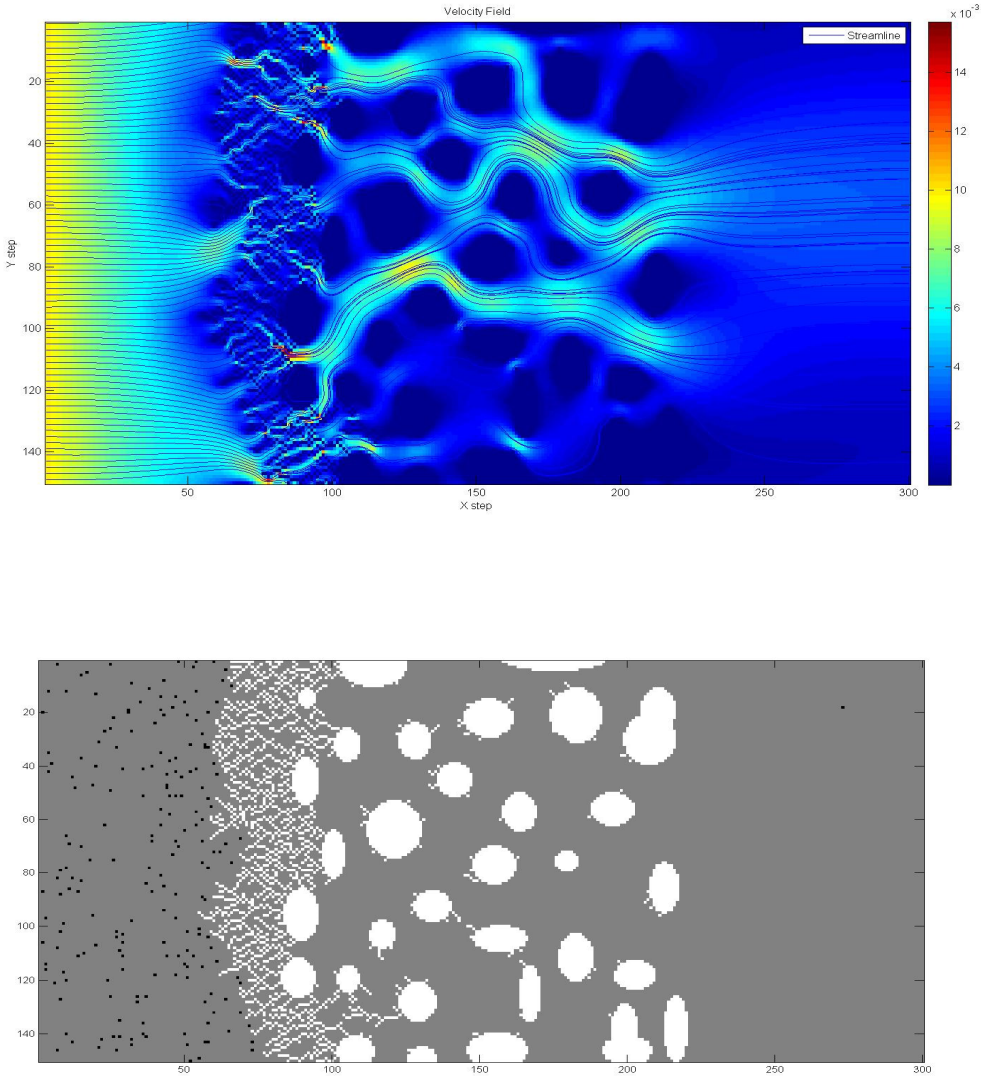


Fig.6.7 Velocity field and deposition pattern using DDA without artificial Brownian motion or shear stress

In the simulation, the filter efficiency obtained for Fig.6.7 is 95.5%. The filter efficiency would grow as the particles would be collected completely according to this algorithm.

The pressure drop is shown in Fig.6.8.

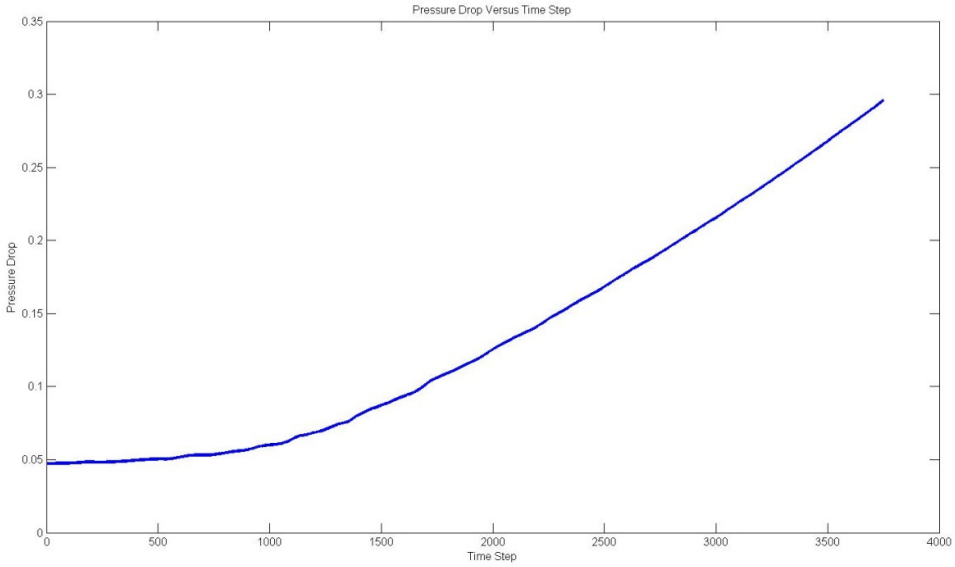
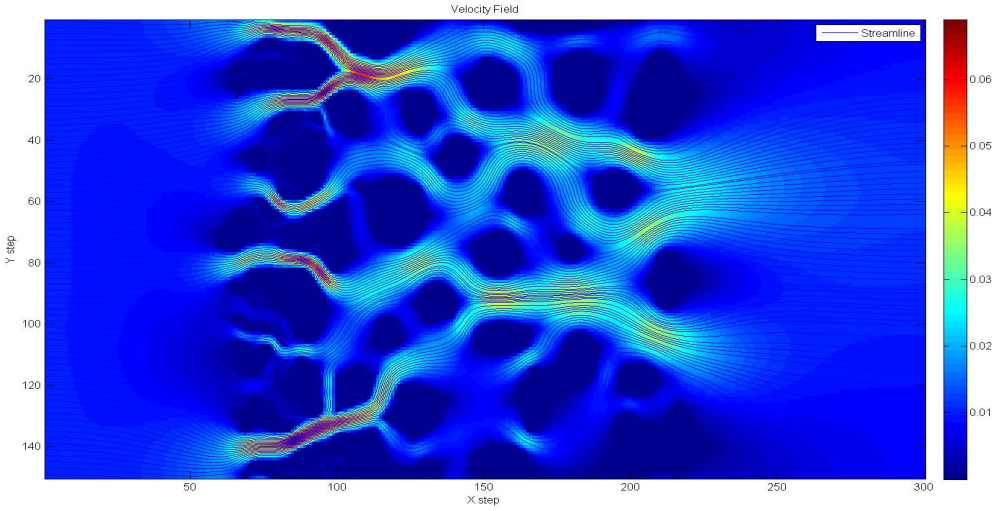


Fig.6.8 Pressure Drop using DDA without artificial Brownian motion or shear stress rule

Then we apply the artificial Brownian motion and shear stress rule to above deposition procedure. The results are presented in Fig.6.9.



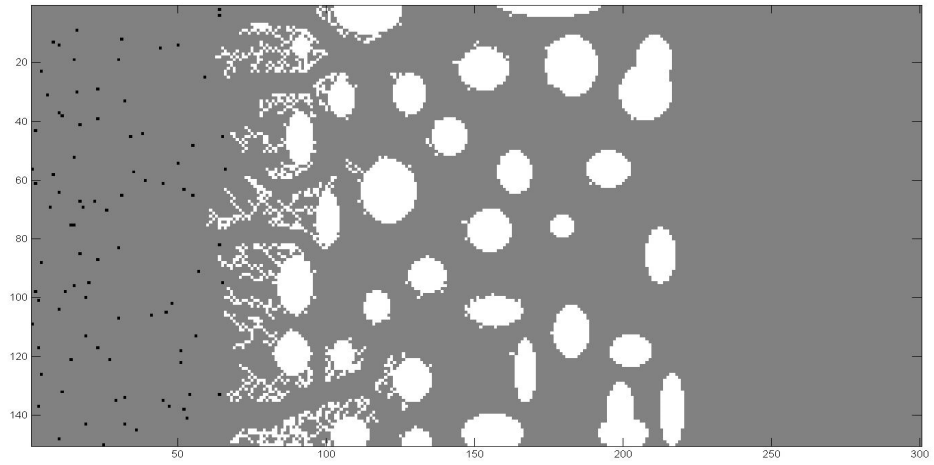


Fig.6.9 Velocity field and deposition pattern using DDA with artificial Brownian motion or shear stress rule

The filter efficiency for this case is 98.4%. The filter efficiency increases by adding the artificial Brownian diffusion.

The pressure drop is given in Fig.6.10.

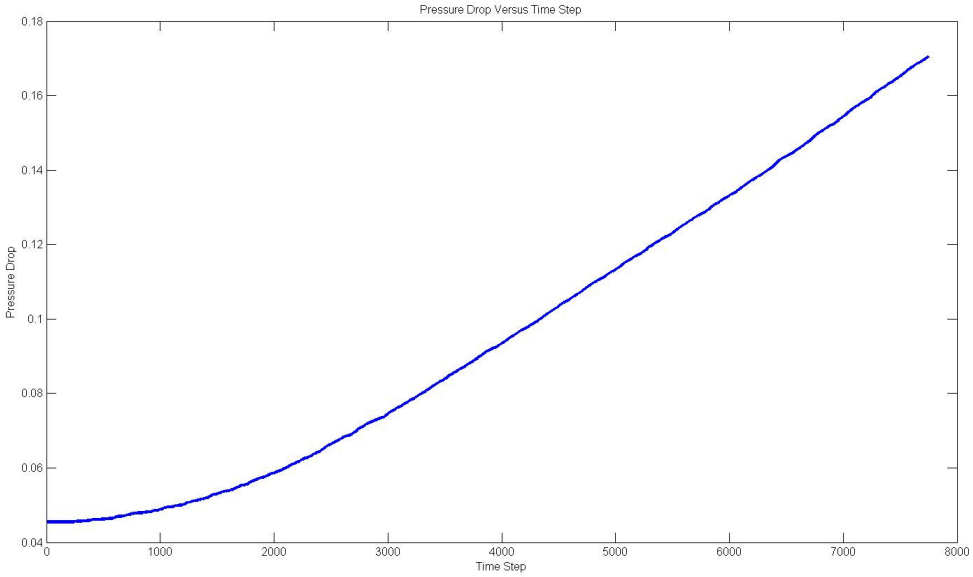


Fig.6.10 Pressure drop using DDA without artificial Brownian motion or shear stress rule

Chapter 7

Conclusion

Throughout this whole paper, the LBM scheme is successfully implemented to simulate the flow field in complex geometry. We implement the particle motion and deposition part with CA. Even though the details of particles are not included, as an approximation method, CA coupled with LBM give satisfactory results.

For the future perspective, there are several points regarding the particle motion and deposition model that need improvement.

The particle motion in the Boolean model is conjugated with a diffusion deviation, which is not isotropic in every direction. Therefore, the artificial Brownian motion needs to be modified to counterbalance the anisotropic effects and eventually fit the real case. However, this part engages with the theoretical part of deriving the Brownian motion, which is not that easy to implement.

The particle size problem is the second difficult one that needs to be implemented. If a rule of CA could be applied to handle this issue, it may be the best way. But this is just an ideal case. Usually more physics related would be introduced.

After all above issues being solved, the final step is to apply this model to the real case. The single fiber collection efficiency is a good criterion to test the model. The model for packed bed should fits this model too.

References

Bao J. Yuan P. Schaefer L. 2008. A mass conserving boundary condition for the lattice Boltzmann equation method. Journal of Computational Physics 227 (2008) 8472–8487.

Bird G.A. 1994. Molecular gas dynamics and the direct simulation of gas flow. Clarendon, Oxford.

Bhatnagar P.L. Gross E.P. Krook M. 1954. A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems. Physical Review. Volume 94. Number 3.

Bouzidi M. Firdaouss M. Lallemand P. 2001. Momentum transfer of a Boltzmann-lattice fluid with boundaries. Phys. Fluids 13, 3452.

Chen S. Doolen G. 1998. Lattice Boltzmann method for fluid flows. Annu. Rev. Fluid Mech. 1998. 30:329–64.

Chen S. Martínez D. Mei R. 1996. On boundary conditions in lattice Boltzmann methods. Phys. Fluids 8, 2527.

Chopard B. Masselot A. 1999. Cellular automata and lattice Boltzmann methods: a new approach to computational fluid dynamics and particle transport. Future Generation Computer Systems 16.249-257.

Chopard B. Masselot A. Dupuis A. 2000. A lattice gas model for erosion and particles transport in a fluid. Computer Physics Communications 129 (2000) 167–176.

Dupuis A. 2002. From a lattice Boltzmann model to a parallel and reusable implementation of a

virtual river. Ph.D. Thesis. Geneva.

Ermak D. Buckholz H. 1980. Numerical Integration of the Langevin Equation: Monte Carlo Simulation. Journal of Computational Physics. 35, 169-l 82 (1980)

Filippova O. Hanel D. 1998. Grid Refinement for Lattice-BGK Models. Journal of Computational Physics 147, 219–228.

Frenklach M. 2002. Method of moments with interpolative closure. Chemical Engineering Science 57 (2002) 2229-2239.

Friedlander S.K. 2000. Smoke, Dust, and Haze, Oxford University Press, New York.

Frisch U. hasslacher B. Pomeau Y. 1986. Lattice-gas automata for the Navier-Stokes equations. Phys. Rev. Lett. 56, 1505-1508.

Gschaider B. Honeger C. 2006. Soot Particle Deposition within Porous Structures using a Method-of-Moments-Lattice-Boltzmann Approach. Inter Journal for Multiscale Computational Engineering, 4(2)221-232.

Guo Z. Shi B. Wang N. 2000. Lattice BGK Model for Incompressible Navier–Stokes Equation Journal of Computational Physics 165, 288–306.

Guo Z. Zheng C. Shi B. 2001. Non-equilibrium extrapolation method for velocity and pressure boundary conditions in the lattice Boltzmann method. Chinese Phys. 11 (04) 366-09.

Hardy J. Pomeau Y. Pazzis O. 1973. Time evolution of a two-dimensional classical lattice system. Phys. Rev. Lett. 31, 276-279.

Hayashi H. Kubo S. 2008. Computer simulation study on filtration of soot particles in diesel particulate filter. Computers and Mathematics with Applications 55. 1450–1460.

He X. Zou Q. Luo L-S. Dembo M. 1997. Analytic Solutions of Simple Flows and Analysis of Nonslip Boundary Conditions for the Lattice Boltzmann BGK Model. Journal of Statistical Physics, Vol. 87, Nos. 1/2,

He X. Luo L-S. 1997. Lattice Boltzmann Model for the Incompressible Navier-Stokes Equation. Journal of Statistical Physics, Vol. 88, Nos. 3/4.

He X. Luo L-S. 1997(a). A priori derivation of the lattice Boltzmann equation. Physical Review E. Volume 55, Number 6.

He X. Luo L-S. 1997(b). Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann euqtion. Physical Review E. Volume 56, Number 6.

He Y. Wang Y. Li Q. 2009. Lattice Boltzmann Method: Theory and applications. Sciencep. Beijing.

Higuera F.J. Succi S. Benzi R. 1989. Lattice Gas Dynamics with Enhanced Collisions. Europhys. Lett. 9 (4), pp. 345-349.

Higuera F.J. Jimenez J. 1989. Boltzmann Approach to Lattice Gas Simulations. Europhys. Lett. 9 (7), pp. 663-668.

Hoffmann K.A. Chiang S.T. 2000. Computational Fluid Dynamics Volume I. Fourth Edition. Engineering Education System. Wichita.

Huynh C.T. Johnson J.H. Yang S.L. Bagley S.T. Warner J.R. 2003. A One-Dimensional Computational Model for Studying the Filtration and Regeneration Characteristics of a Catalyzed

Wall-Flow Diesel Particulate Filter. SAE paper 2003-01-0841.

Inamuro T. Yoshino M. Ogino F. 1995. A non-slip boundary condition for lattice Boltzmann simulations. Phys. Fluids 7, 2928.

Kadanoff L. 1986. On two levels. Phys. Today 39:7-9.

Kang Q. Lichtner P.C. Janecky D.R. 2010. Lattice Boltzmann Method for Reacting Flows in Porous Media. Adv. Appl. Math. Mech., Vol.2, No.5, pp. 545-563.

Kanki T. Iuchi S. 1973. Poiseuille flow and thermal creep of a rarefied gas between parallel plates. Phys. Fluids 16, 594-599.

Konstandopoulos A. Johnson J. 1989. Wall-Flow Diesel Particulate Filters – Their Pressure Drop and Collection Efficiency. SAE Trans. 98, sec. 3 (J. Engines), pp.625-647.

Konstandopoulos. A.G. 2000. Fundamental Studies of Diesel Particulate Filters: Transient Loading, Regeneration and Aging. SAE paper 2000-01-1016

Kuwabara S. 1959. The Forces Experienced by Randomly Distributed Parallel Circular Cylinders or Spheres in a Viscous Fluid at Small Reynolds Numbers, J. Phys. Soc. Japan 14, pp. 527-532.

Ladd A.J.C. 1994. Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 1. Theoretical foundation. J. Fluid Mech. (1994), vol. 211, pp. 285-309.

Lee K.W. Gieseke J.A., 1978. Collection of aerosol particles by packed beds. Env. Sci. Tech. 13 (4), p.1761.

Lee K.W. Liu B.Y.H. 1982. Theoretical Study of Aerosol Filtration by Fibrous Filters. Aerosol

Science and Technology. 1:2, 147-161.

Masselot A. 2000. A New numerical approach to snow transport and deposition by wind: a parallel lattice gas model. Ph.D. Thesis.

Mei R. Luo L-S. Shyy W. 1999. An Accurate Curved Boundary Treatment in the Lattice Boltzmann Method. Journal of Computational Physics 155, 307–330.

McNamara G R, Zanetti G. Use of the Boltzmann equation to simulate lattice automata. Physical Review Letters, 1988, 61(20), 2332-2335

Mohamad A.A. 2011. Lattice Boltzmann Method – Fundamentals and Engineering Applications with Computer Codes. Springer.

Przekop R. Moskal A. Gradon L. 2003. Lattice-Boltzmann approach for description of the structure of deposited particulate matter in fibrous filters. Aerosol Science 34 (2003) 133–147.

Qian Y.H. D'Humieres D. Lallemand P. 1992. Lattice BGK Models for Navier-Stokes Equation. Europhys. Lett. 17 (6), pp. 479-484.

Ramarao B. Tien C. Mohan S. 1994. Calculation of single fiber efficiencies for interception and impaction with superposed brownian motion. J. Aerosol Sci., Vol. 25, No. 2, pp. 295-313.

Sali. R.D. 2012. Two Dimensional Lattice Boltzmann Simulation of Fluid Flow through An Idealized Micro-structure of Disordered Media. Michigan Technological University. Master's Report.

Suh S-H. Kang S-K. Chung D-H. Stroud I. 2008. Theory and Design of CNC Systems. Springer-Verlag London.

Sukop M.C. Thorne. D.T. 2006. Lattice Boltzmann Modeling – An introduction for Geoscientists and Engineers. Springer.

Succi S. 2001. The Lattice Boltzmann Equation for Fluid Dynamics and Beyond. Clarendon Press, Oxford.

Verschaeve J.C.G. 2009. Analysis of the lattice Boltzmann Bhatnagar-Gross-Krook no-slip boundary condition: Ways to improve accuracy and stability. Physical Review E 80, 036703.

Wolf-Gladrow D. 2005. Lattice-Gas Cellular Automata and Lattice Boltzmann models - An introduction. Springer.

Wolfram S. 1983. Statistical mechanics of cellular automata. Rev. Mod. Phys., 55:601-644.

Wolfram S. 1984. Universality and complexity in cellular automata. Physica D, 10:1-35, Porous Media

Yamamoto K. Satake S. Yamashita H. Takada N. Misawa M. 2006. Lattice Boltzmann simulation on porous structure and soot accumulation. Mathematics and Computers in Simulation. 72(2006) 257-263.

Yamamoto K. Satake S. Ymashita H. 2009. Microstructure and particle-laden flow in diesel particulate filter. International Jounal of Thermal Sciences 48 (2009) 303-307.

Yin X. Zhang J. 2012. An improved bounce-back scheme for complex boundary conditions in lattice Boltzmann method. Journal of Computational Physics 231 (2012) 4295–4303.

Yu D. Mei R, Luo L-S, Shyy W. 2003. Viscous flow computations with the method of lattice Boltzmann equation. Progress in Aerospace Sciences 39 (2003) 329–367.

Zou Q. He X. 1997. On pressure and velocity boundary conditions for the lattice Boltzmann BGK model. Phys. Fluids 9, 1591.

Appendix I. Finite-Difference Recovery of Navier-Stokes Equation for Incompressible LBGK Model

In this section, the D2Q9 (two dimensional nine speeds) lattice LBGK model:

$$f_\alpha\left(\mathbf{x}+\mathbf{e}_\alpha\delta_t, t+\delta_t\right)-f_\alpha\left(\mathbf{x},t\right)=-\frac{1}{\tau}\left[f_\alpha\left(\mathbf{x},t\right)-f_\alpha^{(eq)}\left(\rho,\mathbf{u}\right)\right]+\frac{\delta_t^2}{\delta_x}g_\alpha, \tag{A1.1}$$

where c is the lattice speed $c=\delta_x/\delta_t$; δ_x and δ_t are the lattice constant in length and step size in time, τ is the dimensionless relaxation time which is closely relative to Reynolds number, and

$$\mathbf{e}_\alpha=\begin{cases}(0,0) & \alpha=0\\ (\cos[(\alpha-1)\pi/2],\sin[(\alpha-1)\pi/2])c & \alpha=1,2,3,4\\ (\cos[(\alpha-5)\pi/2+\pi/4],\sin[(\alpha-5)\pi/2+\pi/4])\sqrt{2}c & \alpha=5,6,7,8\end{cases}$$

are the velocity vectors.

The equilibrium distribution function $f_\alpha^{(eq)}$ is

$$f_\alpha^{(eq)}\left(\rho,\mathbf{u}\right)=w_\alpha\rho\left[1+3\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c}\right)+\frac{9}{2}\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c^2}\right)^2-\frac{3}{2}\left(\frac{\mathbf{u}}{c}\right)^2\right], \tag{A1.2}$$

Where the weighting factor $w_0=4/9$, $w_{1,2,3,4}=1/9$, $w_{5,6,7,8}=1/36$.

The density of each lattice unit and local momentum is given by

$$\rho=\sum_\alpha f_\alpha; \tag{A1.3}$$

$$\rho\mathbf{u}=\sum_\alpha \mathbf{e}_\alpha f_\alpha; \tag{A1.4}$$

The forcing term is given by

$$g_\alpha=\begin{cases}0 & \alpha=0\\ \frac{1}{3c}\mathbf{e}_\alpha\cdot\mathbf{F} & \alpha=1,2,3,4\\ \frac{1}{12c}\mathbf{e}_\alpha\cdot\mathbf{F} & \alpha=5,6,7,8\end{cases} \tag{A1.5}$$

In finite difference method, the position we study is represented by (i,j), the x and y direction coordinates. At the time step n (before the collision), the distribution function f_α at (i,j), is represented by $f_\alpha^{(n)}(i,j)$. And we can assume that the time step after streaming is represented by superscripts k, and the time step after collision (bounce-back) by $n+k/2$ where the notation k

is integer $(k = 0, 1, 2, 3, \cdots)$.

Therefore, after streaming we have

$$f_0^{(n)}(i,j) = f_0^{\left(n-\frac{1}{2}\right)}(i,j); \tag{A1.6a}$$

$$f_1^{(n)}(i,j) = f_1^{\left(n-\frac{1}{2}\right)}(i-1,j); \tag{A1.6b}$$

$$f_2^{(n)}(i,j) = f_2^{\left(n-\frac{1}{2}\right)}(i,j-1); \tag{A1.6c}$$

$$f_3^{(n)}(i,j) = f_3^{\left(n-\frac{1}{2}\right)}(i+1,j); \tag{A1.6d}$$

$$f_4^{(n)}(i,j) = f_4^{\left(n-\frac{1}{2}\right)}(i,j+1); \tag{A1.6e}$$

$$f_5^{(n)}(i,j) = f_5^{\left(n-\frac{1}{2}\right)}(i-1,j-1); \tag{A1.6f}$$

$$f_6^{(n)}(i,j) = f_6^{\left(n-\frac{1}{2}\right)}(i+1,j-1); \tag{A1.6g}$$

$$f_7^{(n)}(i,j) = f_7^{\left(n-\frac{1}{2}\right)}(i+1,j+1); \tag{A1.6h}$$

$$f_8^{(n)}(i,j) = f_8^{\left(n-\frac{1}{2}\right)}(i-1,j+1). \tag{A1.6i}$$

By equation (1.3) and (1.4), we have

$$u_{i,j}^{(n)} = \frac{c}{\sum_\alpha f_\alpha^{(n)}}\left[\left(f_1^{(n)}(i,j) - f_3^{(n)}(i,j)\right) + \left(f_5^{(n)}(i,j) - f_6^{(n)}(i,j)\right) + \left(f_8^{(n)}(i,j) - f_7^{(n)}(i,j)\right)\right]. \tag{A1.7}$$

Then

$$u_{i,j}^{(n)} = \frac{c}{\rho}\left[\left(f_1^{(n)}(i,j) - f_3^{(n)}(i,j)\right) + \left(f_5^{(n)}(i,j) - f_6^{(n)}(i,j)\right) + \left(f_8^{(n)}(i,j) - f_7^{(n)}(i,j)\right)\right]. \tag{A1.8a}$$

$$v_{i,j}^{(n)} = \frac{c}{\rho}\left[\left(f_2^{(n)}(i,j) - f_4^{(n)}(i,j)\right) + \left(f_5^{(n)}(i,j) - f_8^{(n)}(i,j)\right) + \left(f_6^{(n)}(i,j) - f_7^{(n)}(i,j)\right)\right]. \tag{A1.8b}$$

According to the equations (A1.2) and (A1.6), we may have the distribution function after collision below

$$f_0^{\left(n+\frac{1}{2}\right)}(i,j) = \frac{4}{9\tau}\rho\left[1 - \frac{3}{2}\left(\frac{u_{i,j}^{(n)2} + v_{i,j}^{(n)2}}{c^2}\right)\right] + \frac{\tau-1}{\tau}f_0^{(n)}(i,j); \tag{A1.9a}$$

$$f_1^{\left(n+\frac{1}{2}\right)}(i,j) = \frac{\rho}{9\tau}\left[1 + 3\frac{u_{i,j}^{(n)}}{c} + 3\frac{u_{i,j}^{(n)2}}{c^2} - \frac{3}{2}\frac{v_{i,j}^{(n)2}}{c^2}\right] + \frac{\tau-1}{\tau}f_1^{(n)}(i,j); \tag{A1.9b}$$

$$f_2^{\left(n+\frac{1}{2}\right)}(i,j)=\frac{\rho}{9\tau}\left[1+3\frac{v_{i,j}^{(n)}}{c}-\frac{3}{2}\frac{u_{i,j}^{(n)2}}{c^2}+3\frac{v_{i,j}^{(n)2}}{c^2}\right]+\frac{\tau-1}{\tau}f_2^{(n)}(i,j); \tag{A1.9c}$$

$$f_3^{\left(n+\frac{1}{2}\right)}(i,j)=\frac{\rho}{9\tau}\left[1-3\frac{u_{i,j}^{(n)}}{c}+3\frac{u_{i,j}^{(n)2}}{c^2}-\frac{3}{2}\frac{v_{i,j}^{(n)2}}{c^2}\right]+\frac{\tau-1}{\tau}f_3^{(n)}(i,j); \tag{A1.9d}$$

$$f_4^{\left(n+\frac{1}{2}\right)}(i,j)=\frac{\rho}{9\tau}\left[1-3\frac{v_{i,j}^{(n)}}{c}-\frac{3}{2}\frac{u_{i,j}^{(n)2}}{c^2}+3\frac{v_{i,j}^{(n)2}}{c^2}\right]+\frac{\tau-1}{\tau}f_4^{(n)}(i,j); \tag{A1.9e}$$

$$f_5^{\left(n+\frac{1}{2}\right)}(i,j)=\frac{\rho}{36\tau}\left[1+3\frac{u_{i,j}^{(n)}}{c}+3\frac{v_{i,j}^{(n)}}{c}+3\frac{u_{i,j}^{(n)2}}{c^2}+3\frac{v_{i,j}^{(n)2}}{c^2}+9\frac{u_{i,j}^{(n)}v_{i,j}^{(n)}}{c^2}\right]+\frac{\tau-1}{\tau}f_5^{(n)}(i,j); \tag{A1.9f}$$

$$f_6^{\left(n+\frac{1}{2}\right)}(i,j)=\frac{\rho}{36\tau}\left[1-3\frac{u_{i,j}^{(n)}}{c}+3\frac{v_{i,j}^{(n)}}{c}+3\frac{u_{i,j}^{(n)2}}{c^2}+3\frac{v_{i,j}^{(n)2}}{c^2}-9\frac{u_{i,j}^{(n)}v_{i,j}^{(n)}}{c^2}\right]+\frac{\tau-1}{\tau}f_6^{(n)}(i,j); \tag{A1.9g}$$

$$f_7^{\left(n+\frac{1}{2}\right)}(i,j)=\frac{\rho}{36\tau}\left[1-3\frac{u_{i,j}^{(n)}}{c}-3\frac{v_{i,j}^{(n)}}{c}+3\frac{u_{i,j}^{(n)2}}{c^2}+3\frac{v_{i,j}^{(n)2}}{c^2}+9\frac{u_{i,j}^{(n)}v_{i,j}^{(n)}}{c^2}\right]+\frac{\tau-1}{\tau}f_7^{(n)}(i,j); \tag{A1.9h}$$

$$f_8^{\left(n+\frac{1}{2}\right)}(i,j)=\frac{\rho}{36\tau}\left[1+3\frac{u_{i,j}^{(n)}}{c}-3\frac{v_{i,j}^{(n)}}{c}+3\frac{u_{i,j}^{(n)2}}{c^2}+3\frac{v_{i,j}^{(n)2}}{c^2}-9\frac{u_{i,j}^{(n)}v_{i,j}^{(n)}}{c^2}\right]+\frac{\tau-1}{\tau}f_8^{(n)}(i,j); \tag{A1.9i}$$

Substituting equation set (A1.6) and (A1.9) into (1.8a) gives:

$$\begin{aligned}
u_{i,j}^{(n+1)}=\frac{c}{\rho}\Bigg[&\frac{\rho}{9\tau}\left(1+3\frac{u_{i-1,j}^{(n)}}{c}+3\frac{u_{i-1,j}^{(n)\ 2}}{c^2}-\frac{3}{2}\frac{v_{i-1,j}^{(n)\ 2}}{c^2}\right)+\frac{\tau-1}{\tau}f_1^{(n)}(i-1,j)\\
&-\frac{\rho}{9\tau}\left(1-3\frac{u_{i+1,j}^{(n)}}{c}+3\frac{u_{i+1,j}^{(n)\ 2}}{c^2}-\frac{3}{2}\frac{v_{i+1,j}^{(n)\ 2}}{c^2}\right)-\frac{\tau-1}{\tau}f_3^{(n)}(i+1,j)\\
&+\frac{\rho}{36\tau}\left(1+3\frac{u_{i-1,j-1}^{(n)}}{c}+3\frac{v_{i-1,j-1}^{(n)}}{c}+3\frac{u_{i-1,j-1}^{(n)\ 2}}{c^2}+3\frac{v_{i-1,j-1}^{(n)\ 2}}{c^2}+9\frac{u_{i-1,j-1}^{(n)}v_{i-1,j-1}^{(n)}}{c^2}\right)+\frac{\tau-1}{\tau}f_5^{(n)}(i-1,j-1)\\
&-\frac{\rho}{36\tau}\left(1-3\frac{u_{i+1,j-1}^{(n)}}{c}+3\frac{v_{i+1,j-1}^{(n)}}{c}+3\frac{u_{i+1,j-1}^{(n)\ 2}}{c^2}+3\frac{v_{i+1,j-1}^{(n)\ 2}}{c^2}-9\frac{u_{i+1,j-1}^{(n)}v_{i+1,j-1}^{(n)}}{c^2}\right)-\frac{\tau-1}{\tau}f_6^{(n)}(i+1,j-1)\\
&+\frac{\rho}{36\tau}\left(1+3\frac{u_{i-1,j+1}^{(n)}}{c}-3\frac{v_{i-1,j+1}^{(n)}}{c}+3\frac{u_{i-1,j+1}^{(n)\ 2}}{c^2}+3\frac{v_{i-1,j+1}^{(n)\ 2}}{c^2}-9\frac{u_{i-1,j+1}^{(n)}v_{i-1,j+1}^{(n)}}{c^2}\right)+\frac{\tau-1}{\tau}f_8^{(n)}(i-1,j+1)\\
&-\frac{\rho}{36\tau}\left(1-3\frac{u_{i+1,j+1}^{(n)}}{c}-3\frac{v_{i+1,j+1}^{(n)}}{c}+3\frac{u_{i+1,j+1}^{(n)\ 2}}{c^2}+3\frac{v_{i+1,j+1}^{(n)\ 2}}{c^2}+9\frac{u_{i+1,j+1}^{(n)}v_{i+1,j+1}^{(n)}}{c^2}\right)-\frac{\tau-1}{\tau}f_7^{(n)}(i+1,j+1)\Bigg].
\end{aligned} \tag{A1.10}$$

After expansion and combination,

$$\frac{\rho}{c}u_{i,j}^{(n+1)}=\frac{\rho}{36\tau}\Bigg(12\frac{u_{i-1,j}^{(n)}+u_{i+1,j}^{(n)}}{c}+12\frac{{u_{i-1,j}^{(n)}}^2-{u_{i+1,j}^{(n)}}^2}{c^2}-6\frac{{v_{i-1,j}^{(n)}}^2-{v_{i+1,j}^{(n)}}^2}{c^2}$$

$$+3\frac{u_{i-1,j-1}^{(n)}+u_{i+1,j-1}^{(n)}}{c}+3\frac{v_{i-1,j-1}^{(n)}-v_{i+1,j-1}^{(n)}}{c}+3\frac{{u_{i-1,j-1}^{(n)}}^2-{u_{i+1,j-1}^{(n)}}^2}{c^2}+3\frac{{v_{i-1,j-1}^{(n)}}^2-{v_{i+1,j-1}^{(n)}}^2}{c^2}$$

$$+9\frac{u_{i-1,j-1}^{(n)}v_{i-1,j-1}^{(n)}+u_{i+1,j-1}^{(n)}v_{i+1,j-1}^{(n)}}{c^2}$$

$$+3\frac{u_{i-1,j+1}^{(n)}+u_{i+1,j+1}^{(n)}}{c}-3\frac{v_{i-1,j+1}^{(n)}-v_{i+1,j+1}^{(n)}}{c}+3\frac{{u_{i-1,j+1}^{(n)}}^2-{u_{i+1,j+1}^{(n)}}^2}{c^2}+3\frac{{v_{i-1,j+1}^{(n)}}^2-{v_{i+1,j+1}^{(n)}}^2}{c^2}$$

$$-9\frac{u_{i-1,j+1}^{(n)}v_{i-1,j+1}^{(n)}+u_{i+1,j+1}^{(n)}v_{i+1,j+1}^{(n)}}{c^2}\Bigg)$$

$$+\frac{\tau-1}{\tau}\Big[f_1^{(n)}(i-1,j)-f_3^{(n)}(i+1,j)+f_5^{(n)}(i-1,j-1)-f_6^{(n)}(i+1,j-1)+f_8^{(n)}(i-1,j+1)$$

$$-f_7^{(n)}(i+1,j+1)\Big];\qquad\text{(A1.11)}$$

Then

$$u_{i,j}^{(n+1)}=\frac{1}{12\tau}\Bigg[\ 4\left(u_{i-1,j}^{(n)}+u_{i+1,j}^{(n)}\right)+4\frac{{u_{i-1,j}^{(n)}}^2-{u_{i+1,j}^{(n)}}^2}{c}-2\frac{{v_{i-1,j}^{(n)}}^2-{v_{i+1,j}^{(n)}}^2}{c}$$

$$+\left(u_{i-1,j-1}^{(n)}+u_{i+1,j-1}^{(n)}\right)+\left(v_{i-1,j-1}^{(n)}-v_{i+1,j-1}^{(n)}\right)+\frac{{u_{i-1,j-1}^{(n)}}^2-{u_{i+1,j-1}^{(n)}}^2}{c}+\frac{{v_{i-1,j-1}^{(n)}}^2-{v_{i+1,j-1}^{(n)}}^2}{c}$$

$$+\left(u_{i-1,j+1}^{(n)}+u_{i+1,j+1}^{(n)}\right)-\left(v_{i-1,j+1}^{(n)}-v_{i+1,j+1}^{(n)}\right)+\frac{{u_{i-1,j+1}^{(n)}}^2-{u_{i+1,j+1}^{(n)}}^2}{c}+\frac{{v_{i-1,j+1}^{(n)}}^2-{v_{i+1,j+1}^{(n)}}^2}{c}$$

$$+3\frac{u_{i-1,j-1}^{(n)}v_{i-1,j-1}^{(n)}+u_{i+1,j-1}^{(n)}v_{i+1,j-1}^{(n)}-u_{i-1,j+1}^{(n)}v_{i-1,j+1}^{(n)}-u_{i+1,j+1}^{(n)}v_{i+1,j+1}^{(n)}}{c}\Bigg]$$

$$+\frac{c}{\rho}\frac{\tau-1}{\tau}\Big[f_1^{(n)}(i-1,j)-f_3^{(n)}(i+1,j)+f_5^{(n)}(i-1,j-1)-f_6^{(n)}(i+1,j-1)+f_8^{(n)}(i-1,j+1)$$

$$-f_7^{(n)}(i+1,j+1)\Big];\qquad\text{(A1.12)}$$

Through Taylor series expansion, the specific details can be consulted in the book [Hoffmann & Chiang 2000], then we have

$$4\left(u_{i-1,j}^{(n)}+u_{i+1,j}^{(n)}\right)=4\left(2u_{i,j}^{(n)}+\delta_x^2\frac{\partial^2 u}{\partial x^2}+O\left[(\delta_x)^4\right]\right)$$

$$u_{i-1,j-1}^{(n)}+u_{i+1,j-1}^{(n)}+u_{i-1,j+1}^{(n)}+u_{i+1,j+1}^{(n)}=4u_{i,j}^{(n)}+2\delta_x^2\frac{\partial^2 u}{\partial x^2}+2\delta_y^2\frac{\partial^2 u}{\partial y^2}+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

$$v_{i-1,j-1}^{(n)}-v_{i+1,j-1}^{(n)}-v_{i-1,j+1}^{(n)}+v_{i+1,j+1}^{(n)}=4\delta_x\delta_y\frac{\partial^2 v}{\partial x\partial y}+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

$$4\left(u_{i-1,j}^{(n)\,2}-u_{i+1,j}^{(n)\,2}\right)=4\left[-2\delta_x\frac{\partial\left(u^2\right)}{\partial x}\right]+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

$$-2\left(v_{i-1,j}^{(n)\,2}-v_{i+1,j}^{(n)\,2}\right)=-2\left[-2\delta_x\frac{\partial\left(v^2\right)}{\partial x}\right]+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

$$u_{i-1,j-1}^{(n)\,2}-u_{i+1,j-1}^{(n)\,2}=-2\delta_x\frac{\partial\left(u^2\right)}{\partial x}+2\delta_x\delta_y\frac{\partial^2\left(u^2\right)}{\partial x\partial y}+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

$$u_{i-1,j+1}^{(n)\,2}-u_{i+1,j+1}^{(n)\,2}=-2\delta_x\frac{\partial\left(u^2\right)}{\partial x}-2\delta_x\delta_y\frac{\partial^2\left(u^2\right)}{\partial x\partial y}+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

$$v_{i-1,j-1}^{(n)\,2}-v_{i+1,j-1}^{(n)\,2}+v_{i-1,j+1}^{(n)\,2}-v_{i+1,j+1}^{(n)\,2}=-4\delta_x\frac{\partial\left(v^2\right)}{\partial x}+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

$$3\left(u_{i-1,j-1}^{(n)}v_{i-1,j-1}^{(n)}+u_{i+1,j-1}^{(n)}v_{i+1,j-1}^{(n)}-u_{i-1,j+1}^{(n)}v_{i-1,j+1}^{(n)}-u_{i+1,j+1}^{(n)}v_{i+1,j+1}^{(n)}\right)=-12\delta_y\frac{\partial(uv)}{\partial y}+O\left[(\delta_x)^3,(\delta_y)^3\right];$$

Substituting all above equations and $c=\delta_x/\delta_t$, $\delta_y=\delta_x$ into A1.12gives

$$u_{i,j}^{(n+1)}=\frac{1}{12\tau}\Bigg[8u_{i,j}^{(n)}+4\delta_x^2\frac{\partial^2 u}{\partial x^2}-8\delta_t\frac{\partial\left(u^2\right)}{\partial x}+4\delta_t\frac{\partial\left(v^2\right)}{\partial x}+4u_{i,j}^{(n)}+2\delta_x^2\frac{\partial^2 u}{\partial x^2}+2\delta_x^2\frac{\partial^2 u}{\partial y^2}+4\delta_x^2\frac{\partial^2 v}{\partial x\partial y}$$

$$-4\delta_t\frac{\partial\left(u^2\right)}{\partial x}-4\delta_t\frac{\partial\left(v^2\right)}{\partial x}-12\delta_t\frac{\partial(uv)}{\partial y}\Bigg]+O\left[(\delta_x)^2,(\delta_y)^2\right]+\frac{c}{\rho}\frac{\tau-1}{\tau}\Big[f_1^{(n)}(i-1,j)-f_3^{(n)}(i+1,j)$$

$$+f_5^{(n)}(i-1,j-1)-f_6^{(n)}(i+1,j-1)+f_8^{(n)}(i-1,j+1)-f_7^{(n)}(i+1,j+1)\Big]$$

When $\tau=1$, using Forward Time Central Space (FTCS) approximation, the equation gives

$$\frac{\partial u}{\partial t}=\frac{1}{12}\left[\frac{6\delta_x^2}{\delta_t}\frac{\partial^2 u}{\partial x^2}+\frac{2\delta_x^2}{\delta_t}\frac{\partial^2 u}{\partial y^2}-12\frac{\partial\left(u^2\right)}{\partial x}+\frac{4\delta_x^2}{\delta_t}\frac{\partial^2 v}{\partial x\partial y}-12\frac{\partial(uv)}{\partial y}\right];$$

Rearranging

$$\frac{\partial u}{\partial t}+\frac{\partial\left(u^2\right)}{\partial x}+\frac{\partial(uv)}{\partial y}=\frac{1}{12}\left[\frac{6\delta_x^2}{\delta_t}\frac{\partial^2 u}{\partial x^2}+\frac{2\delta_x^2}{\delta_t}\frac{\partial^2 u}{\partial y^2}+\frac{4\delta_x^2}{\delta_t}\frac{\partial^2 v}{\partial x\partial y}\right];$$

Recall the continuity equation assuming the density is constant,

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0.$$

Which indicates

$$\frac{\partial^2 v}{\partial x\partial y}=\frac{\partial}{\partial x}\left(\frac{\partial v}{\partial y}\right)=\frac{\partial}{\partial x}\left(-\frac{\partial u}{\partial x}\right)=-\frac{\partial^2 u}{\partial x^2}$$

Then the FTCS approximation gives

$$\frac{\partial u}{\partial t}+\frac{\partial\left(u^2\right)}{\partial x}+\frac{\partial(uv)}{\partial y}=\frac{1}{6}\frac{\delta_x^2}{\delta_t}\left[\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right]$$

The viscosity given by

$$\nu=\frac{2\tau-1}{6}\frac{\delta_x^2}{\delta_t}.$$

The pressure term is coupled with the distribution function through equation of state. The above equation is the N-S equation for the incompressible flow. The continuity equation in the LBGK model can also be demonstrated in this way. This recovery of N-S equation is only valid in the special case $\tau=1$. For the case $\tau\neq 1$, we can recall the successive over relaxation (SOR) method used in CFD.

Appendix II. Incompressible Models

He & Luo's Model

He & Luo proposed an incompressible model in 1997 [He & Luo 1997], in their paper, the author change the original equilibrium density distribution function to:

$$f_\alpha^{(eq)}(\rho,\mathbf{u}) = w_\alpha\left\{\rho+\rho_0\left[3\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c}\right)+\frac{9}{2}\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c^2}\right)^2-\frac{3}{2}\left(\frac{\mathbf{u}}{c}\right)^2\right]\right\};$$

And the authors demonstrate that through this transformation, the remaining error terms are of the order $O(M^2)$ or lower order. This approximation is based on the assumption of low Mach number.

After this step, a local pressure distribution function is introduced:

$$p_\alpha \equiv c_s^2 f_\alpha .$$

where c_s is the sound speed, and $c_s^2 = c^2/3$ for the D2Q9 model. The above equation is also called equation of state and the pressure p_α becomes an independent variable in the incompressible N-S equation.

Then the equation of the system comes to

$$p_\alpha(\mathbf{x}+\mathbf{e}_\alpha\delta_t, t+\delta_t) - p_\alpha(\mathbf{x},t) = -\frac{1}{\tau}\left[p_\alpha(\mathbf{x},t) - p_\alpha^{(eq)}(\rho,\mathbf{u})\right].$$

where

$$p_\alpha^{(eq)}(\rho,\mathbf{u}) \equiv c_s^2 f_\alpha^{(eq)} = w_\alpha\left\{p+p_0\left[3\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c}\right)+\frac{9}{2}\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c^2}\right)^2-\frac{3}{2}\left(\frac{\mathbf{u}}{c}\right)^2\right]\right\}.$$

The only condition must satisfy in numerical simulations of incompressible flow is

$$M \ll 1.$$

Specifically in practice, the value of Mach number is always maintained under 0.15 in numerical simulations by LBE method.

This model is tested in Sali's report [Sali 2012] which shows that it has a quite satisfactory simulation result.

Guo's Model

In 2000, Guo also proposed an incompressible purely based on the mathematical model [Guo et al 2000].

Guo introduced a new equilibrium distribution function $g_i^{(0)}$ defined by

$$g_i^{(0)} = \begin{cases} -4\sigma \dfrac{p}{c^2} + s_0(\mathbf{u}) & i = 0 \\ \lambda \dfrac{p}{c^2} + s_i(\mathbf{u}) & i = 1,2,3,4 \\ \gamma \dfrac{p}{c^2} + s_i(\mathbf{u}) & i = 5,6,7,8 \end{cases}$$

where the parameters satisfying:

$$\lambda + \gamma = \sigma, \quad \lambda + 2\gamma = 1/2 .$$

and

$$s_i(\mathbf{u}) = w_i \left[3\left(\frac{\mathbf{e}_i \cdot \mathbf{u}}{c} \right) + \frac{9}{2}\left(\frac{\mathbf{e}_i \cdot \mathbf{u}}{c^2} \right)^2 - \frac{3}{2}\left(\frac{\mathbf{u}}{c} \right)^2 \right]$$

The weighting factors are same as previous model. Then the evolution equation of the system comes to

$$g_\alpha(\mathbf{x} + \mathbf{e}_\alpha \delta_t, t + \delta_t) - g_\alpha(\mathbf{x}, t) = -\frac{1}{\tau}\left[g_\alpha(\mathbf{x}, t) - g_\alpha^{(eq)}(\rho, \mathbf{u}) \right].$$

The velocity and pressure of flow are given by

$$\mathbf{u} = \sum_{i=1}^{8} c\mathbf{e}_i g_i$$

$$p = \frac{c^2}{4\sigma}\left[\sum_{i=1}^{8} g_i + s_0(\mathbf{u}) \right]$$

This model is widely accepted by other researchers [Kang et al 2010].

Appendix III. Other Boundary Conditions

Mid-Grid Boundary Condition

The picture below shows the scheme of mid-grid bounce back. The empty circles represent nodes inside wall and solid circles represent nodes in the fluid domain. The dashed line symbolizes the location where wall is.

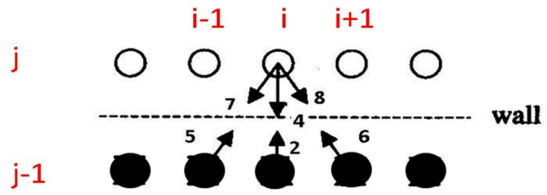


Fig.A3.1 Mid-grid bounce back scheme [Succi 2001]

Succi gives an equation for mid-grid bounce back.

$$\begin{bmatrix} f_7(x,y) \\ f_4(x,y) \\ f_8(x,y) \end{bmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} f_5(x-1,y-1) \\ f_2(x,y-1) \\ f_6(x+1,y-1) \end{bmatrix}.$$

Inamoro Nonslip boundary condition

In Inamuro's paper "A non-slip boundary condition for lattice Boltzmann simulations" [Inamuro et al 1995], it is assumed that gas molecules strike the wall and then leave the wall with a Maxwellian velocity distribution and have the velocity and the temperature of the wall.

For the south wall, he proposes

$$f_2 = \frac{1}{9}\rho'\left\{1+3v_w+\frac{9}{2}{v_w}^2-\frac{3}{2}\left[(u_w+u')^2+{v_w}^2\right]\right\};$$

$$f_5 = \frac{1}{36}\rho'\left\{1+3(u_w+u'+v_w)+\frac{9}{2}(u_w+u'+v_w)^2-\frac{3}{2}\left[(u_w+u')^2+{v_w}^2\right]\right\};$$

$$f_6 = \frac{1}{36}\rho'\left\{1+3(-u_w-u'+v_w)+\frac{9}{2}(-u_w-u'+v_w)^2-\frac{3}{2}\left[(u_w+u')^2+{v_w}^2\right]\right\};$$

where the unknown parameters

$$\rho_w = 1-v_w\left[f_0+f_1+f_3+2(f_4+f_7+f_8)\right];$$

$$\rho' = 6\left[\rho_w v_w + \frac{f_4 + f_7 + f_8}{1 + 3v_w + 3v_w{}^2}\right];$$

$$u' = \frac{1}{1+3v_w}\left\{6\left[\rho_w u_w - \frac{\left(f_1 - f_3 + f_8 - f_7\right)}{\rho'}\right] - u_w - 3u_w v_w\right\}.$$

where u' is the counter-slip velocity.

It is assumed that gas molecules strike the wall leave the wall with a Maxwellian velocity distribution and have the velocity and the temperature of the wall. This method should have 2nd order accuracy but much more complex than other methods. However, it increase the accuracy at the cost of simplicity, which makes this scheme less applicable in programming.

Filippova & Hanel's boundary condition

In 1998, Filippova and Hanel proposed the way of dealing with the curved boundary condition [Filippova O. Hanel D. 1998].

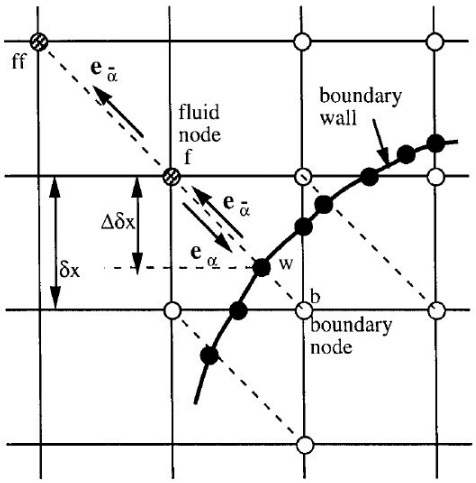


Fig.A3.2 Mei's boundary scheme [Mei et al 1999]

$$\Delta = \left|\frac{\mathbf{x}_f - \mathbf{x}_w}{\mathbf{x}_f - \mathbf{x}_b}\right|;$$

FH proposed the following treatment for $\tilde{f}_{\bar{\alpha}}(\mathbf{x}_b, t)$ on curved boundaries:

$$\tilde{f}_{\bar{\alpha}}(\mathbf{x}_b, t) = (1-\chi)\tilde{f}_{\alpha}(\mathbf{x}_f, t) + \chi f_{\alpha}^{*}(\mathbf{x}_b, t) + 2w_{\alpha}\rho\frac{3}{c^2}\mathbf{e}_{\bar{\alpha}} \cdot \mathbf{u}_w$$

The tilde denotes the post-collision state of distribution function. And $\mathbf{e}_{\bar{\alpha}}$ and $\mathbf{e}_{\alpha}$ denote opposite direction. The χ is the weighting factor, and $f_{\alpha}^{*}(\mathbf{x}_b, t)$ is fictitious equilibrium distribution:

$$f_\alpha^*\left(\mathbf{x}_b,t\right)=w_\alpha\rho\left(\mathbf{x}_f,t\right)\left[1+\frac{3}{c^2}\mathbf{e}_\alpha\cdot\mathbf{u}_{bf}+\frac{9}{2c^4}\left(\mathbf{e}_\alpha\cdot\mathbf{u}_f\right)^2-\frac{3}{2c^2}\mathbf{u}_f\cdot\mathbf{u}_f\right]$$

In above equations, $\mathbf{u}_{bf}$ and χ are to be calculated.

$$\mathbf{u}_{bf}=\frac{(\Delta-1)}{\Delta}\mathbf{u}_f+\frac{1}{\Delta}\mathbf{u}_w,\quad \chi=\frac{(2\Delta-1)}{\tau},\quad for\ \Delta\geq\frac{1}{2}$$
$$\mathbf{u}_{bf}=\mathbf{u}_f,\quad \chi=\frac{(2\Delta-1)}{\tau-1},\qquad for\ \Delta<\frac{1}{2}$$

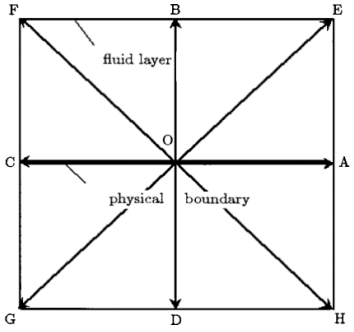
Mei et al improved the stability of this scheme by setting [Mei et al 1999]:

$$\mathbf{u}_{bf}=\left(1-\frac{3}{2\Delta}\right)\mathbf{u}_f+\frac{3}{2\Delta}\mathbf{u}_w,\quad \chi=\frac{(2\Delta-1)}{\tau+\frac{1}{2}},\quad for\ \Delta\geq\frac{1}{2}$$
$$\mathbf{u}_{bf}=\mathbf{u}_{ff},\quad \chi=\frac{(2\Delta-1)}{\tau-2},\qquad for\ \Delta<\frac{1}{2}$$

In Bao's paper, the author points out that using this kind of scheme would cause mass loss or "mass leakage" when body forces are introduced. And the author also propose a mass-conserving boundary condition [Bao et al 2008].

Extrapolation Boundary Condition

The extrapolation scheme is an traditional finite difference method dealing with the boundary condition. And this scheme is introduced to lattice Boltzmann method by Chen et al [Chen et al 1996].



(1) Nodes F, B, E are inside fluid

(0) Nodes C, O, A are on the boundary

(-1) Nodes G, D, H are inside solid

Fig.A3.3 Extrapolation boundary scheme

The scheme gives

$$f_i^{(-1)}=2f_i^{(0)}-f_i^{(1)}$$

This scheme has 2nd order accuracy claimed by the author.

It is tested at $\tau = 0.8$, the result shows

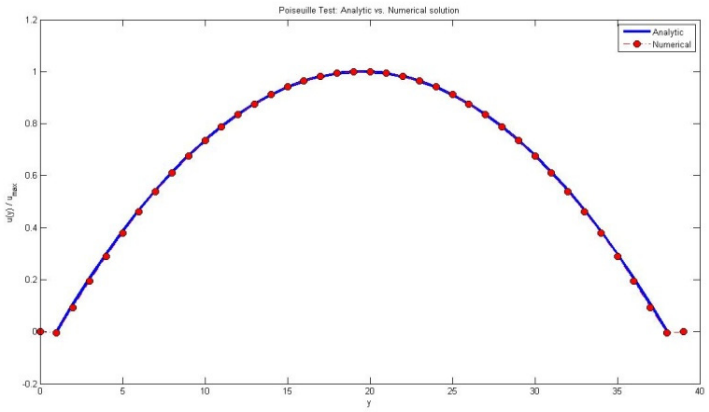


Fig.A3.4 Extrapolation scheme for $\tau = 0.8$

The above figure indicates this scheme might only have 1st order accuracy.

Guo et al [Guo et al 2001] also investigates this extrapolation scheme and modified it in his incompressible model [Guo et al 2000], he also demonstrates that the model have a 2nd order accuracy.

Complex Boundary Condition

A.J.C Ladd proposed the boundary problem regarding hydrodynamic interactions between solid particles [Ladd 1994]. When the boundary geometry is not that simple as straight line, rather cursive or moving boundary, the simple way like bounce-back scheme is far from enough. Since particle size and shape are not uniform, the boundary surface may not right lies on the computational nodes.

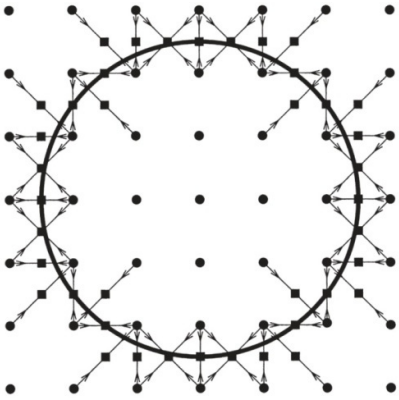


Fig.A3.5 bounce-back [Ladd 1994] (Square Nodes represent practical boundary)

Therefore, there is need to study the special treatment to these types of boundary conditions.

Bouzidi's Boundary Condition

Bouzidi used an interpolation scheme to tackle this type of problem [Bouzidi et al 2001].

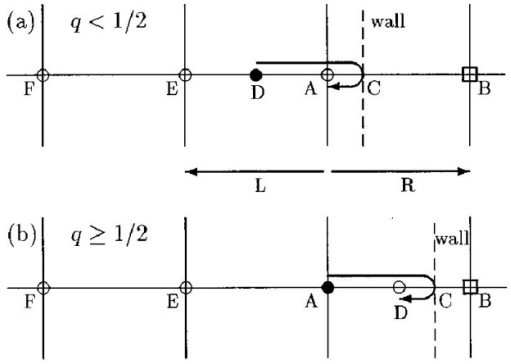


Fig.A3.6 Details of the collision process [Bouzidi 2001]

In above figure, node C represents the exact position of wall, node B is inside wall and node A is in the flow.

Using the linear interpolation:

$$f_{i'}(\mathbf{r}_l, t+1) = 2q f_i^c(\mathbf{r}_l, t) + (1-2q) f_i^c(\mathbf{r}_l - \mathbf{c}_i, t), \quad q < \frac{1}{2}$$

$$f_{i'}(\mathbf{r}_l, t+1) = \frac{1}{2q} f_i^c(\mathbf{r}_l, t) + \frac{(2q-1)}{2q} f_{i'}^c(\mathbf{r}_i, t), \qquad q \geq \frac{1}{2}$$

Where $q = |AC|/|AB|$. The superscript *c* represents distribution function after collision. The prime notation $f_{i'}^c$ represents the reversed direction of f_i^c.

Also they used quadratic interpolation to obtain:

$$f_{i'}(\mathbf{r}_l, t+1) = q(2q+1) f_i^c(\mathbf{r}_l, t) + (1+2q)(1-2q) f_i^c(\mathbf{r}_l - \mathbf{c}_i, t) - q(1-2q) f_i^c(\mathbf{r}_l - 2\mathbf{c}_i, t), \quad q < \frac{1}{2}$$

$$f_{i'}(\mathbf{r}_l, t+1) = \frac{1}{q(2q+1)} f_i^c(\mathbf{r}_l, t) + \frac{(2q-1)}{q} f_{i'}^c(\mathbf{r}_i, t) + \frac{(1-2q)}{(2q+1)} f_{i'}^c(\mathbf{r}_l - \mathbf{c}_i, t), \qquad q \geq \frac{1}{2}$$

It is tested under Poiseuille flow by setting different *q* value

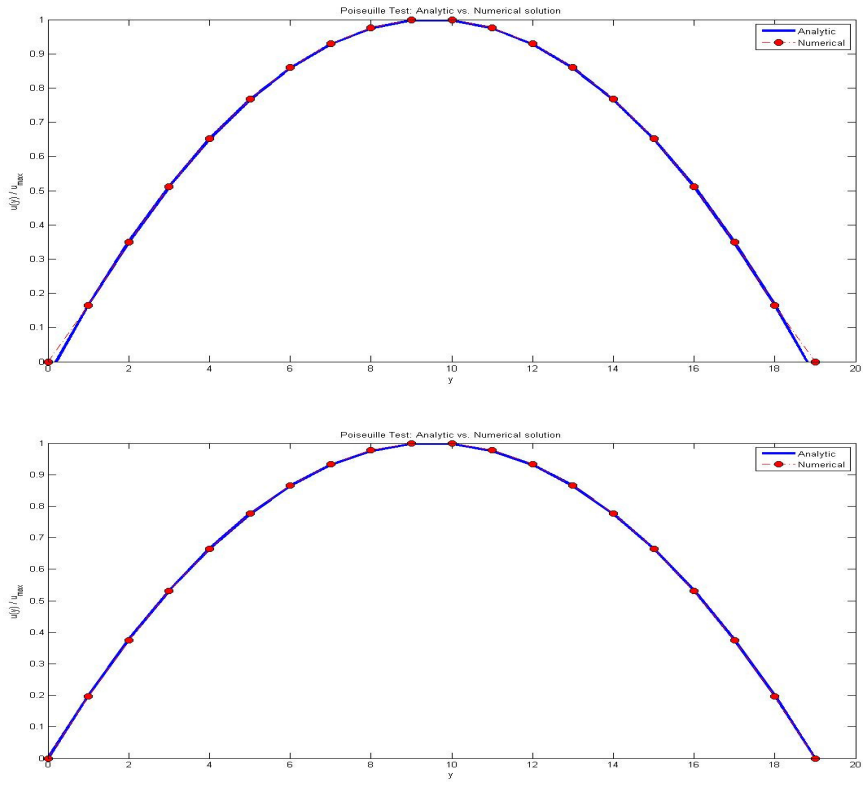


Fig.A3.7 Bouzidi's boundary scheme tested at difference q value

The upper figure is tested at q=0.8 and the lower one at q=1, which shows this scheme has 2nd order accuracy.

In Yin's paper [Yin & Zhang 2012], the author used Bouzidi's idea to interpolate velocity in diagonal direction based on Ladd's paper [Ladd 1994].

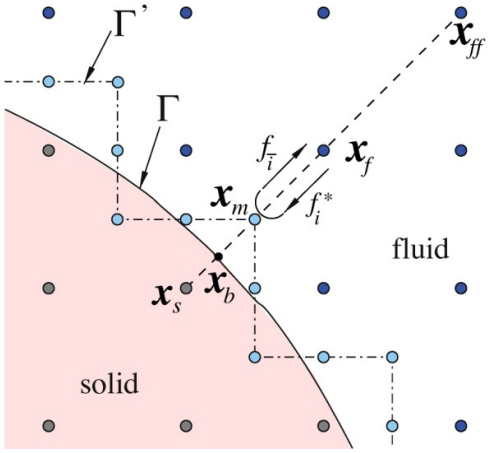


Fig.A3.8 Velocity inter-/extrapolation boundary treatment [Yin 2012]

In above figure, Γ is the solid boundary, Γ' is the solid boundary based on bounce-back scheme. After collision at $\mathbf{x}_f$, the probability density function is assumed to bounce back at midpoint $\mathbf{x}_m$ with modified magnitude:

$f_{i'} = f_i - \frac{2\rho w_i}{c_s^2}\mathbf{u}_m \cdot \mathbf{c}_i$, specifically in case $f_5 = f_7 - \frac{2\rho}{36c_s^2}\mathbf{u}_m \cdot \mathbf{c}_i$.

$$\Delta = \left| \frac{\mathbf{x}_s - \mathbf{x}_b}{\mathbf{x}_s - \mathbf{x}_f} \right|;$$

For $\Delta \le 1/2$, the midpoint velocity $\mathbf{u}_m$ can be obtained with linear interpolation:

$$\mathbf{u}_m = \frac{\frac{1}{2}\mathbf{u}_b + \left(\frac{1}{2} - \Delta\right)\mathbf{u}_f}{1 - \Delta},$$

For $\Delta > 1/2$, the midpoint becomes a solid node, and

$$\mathbf{u}_m = \frac{\frac{3}{2}\mathbf{u}_b + \left(\frac{1}{2} - \Delta\right)\mathbf{u}_{ff}}{2 - \Delta}.$$

We also tested this scheme at the value of q = 0.8 in contrast with the scheme given by Bouzidi.

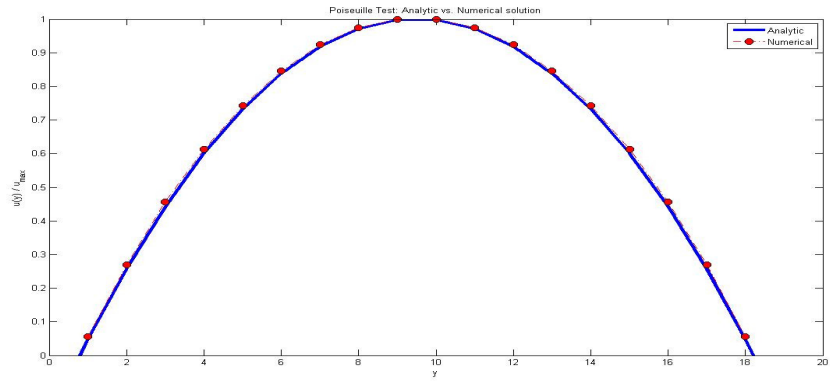


Fig.A3.9 Yin's boundary scheme at q = 0.8.

The result shows that this scheme only has 1st order of accuracy by the tested results.

Appendix IV. Programs

DDA Scheme for Particle Motion

```
u_max = max(max(abs(u)));       % Initialization of maximum velocity
ddax = rand(lx,ly)*u_max;       % Initialization of register for x-direction
dday = rand(lx,ly)*u_max;       % Initialization of register for y-direction
sigx = sign(xu);                % Get the sign of value of velocity
sigy = sign(yu);

[X,Y] = find(P_Area);           % Find the positions of particles
[X,Y] = find(P_Area);           % Find the positions of particles
index = sub2ind(siz,X,Y);    % Transfer the matrix position to index

ddax(index) = ddax(index) + abs(xu(index));      % Adding the x-velocity to register ddax
dday(index) = dday(index) + abs(yu(index));      % Adding the y-velocity to register dday

[X1,Y1] = find(ddax>u_max);                      % Find those registers which are full
index1 = sub2ind(siz,X1,Y1);
[X2,Y2] = find(dday>u_max);
index2 = sub2ind(siz,X2,Y2);

ddax(index1) = ddax(index1) - u_max;             % Clear the registers which are full
dday(index2) = dday(index2) - u_max;

%% Move the particles of which their registers are full (Periodic Boundary condition at upper and
lower walls)
vector_x = zeros(lx,ly);
vector_x(index1) = sigx(index1);
vector_y = zeros(lx,ly);
vector_y(index2) = sigy(index2);
P_Temp = zeros(lx,ly);
for j = 1:lx
    for k = 1:ly
        if((j+vector_x(j,k))>lx || (j+vector_x(j,k))==0)
            vector_x(j,k)=0;
        end
        if((k+vector_y(j,k))>ly)
            vector_y(j,k) = - ly + 1;
        end
        if((k+vector_y(j,k))== 0)
            vector_y(j,k) = ly - 1;
        end
        P_Temp(j+vector_x(j,k),k+vector_y(j,k)) = P_Temp(j+vector_x(j,k),k+vector_y(j,k)) +
```

```
            P_Area(j,k);
        end
end
P_Area = P_Temp;
```

Modified Probability Motion Scheme

```
tau_s = 3;                  % Set the tau_s in order to increase the propagation speed
sigx = sign(xu);            % Get the sign of value of velocity
sigy = sign(yu);

prob_x = tau_s*abs(xu);     % Calculate the probability move along x-direction
prob_y = tau_s*abs(yu);     % Calculate the probability move along y-direction

prob_crit_x = rand(lx,ly);  % Set numbers of uniform distribution range from 0-1
prob_crit_y = rand(lx,ly);

[X1,Y1] = find(prob_x>prob_crit_x);      % Get the positions of particles which will move
index1 = sub2ind(siz,X1,Y1);
[X2,Y2] = find(prob_y>prob_crit_y);
index2 = sub2ind(siz,X2,Y2);
%% The particle movement part is same as that in DDA scheme.
```