Computational System for Visualization and Lattice Boltzmann Fluid Simulation

José Guilherme Mayworm, Sicilia Ferreira Judice and Gilson Antonio Giraldi
National Laboratory for Scientific Computing - Petropolis/RJ, Brazi
Faculty of Technical Education of the State of Rio de Janeiro - Petrópolis/RJ, Brazil
Email: {jmayworm,sicilia,gilson}@lncc.br

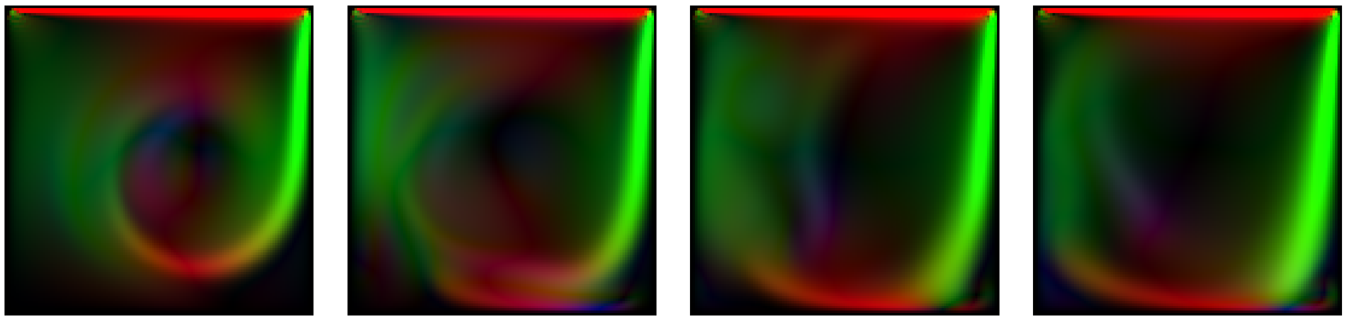


Fig. 1. Visualization of the velocity field of a lid-driven cavity experiment through Lattice Boltzmann fluid simulation.

***Abstract*—This work focus on scientific visualization techniques and fluid simulation via Lattice Boltzmann Method (LBM). The LBM is based on the fundamental idea of constructing simplified kinetic models of fluids, which incorporates the essential physics of microscopic processes so that the macroscopic averaged properties satisfy macroscopic equations. In this work, we developed a fluid simulator based on 3D LBM. We incorporate the simulation kernel in a software with GLUI user interfaces and traditional fluid visualization techniques implemented through OpenGL resources. The simulations can be controlled through the interfaces as well as the parameters of visualization. We present two fluid simulation examples and describe the capabilities of the computational system developed.**

***Keywords*-Fluid Simulation; Lattice Boltzmann; Scientific Visualization; Software Development.**

I. Introduction

The study of fluid flow and its computational simulation is extremely important from the scientific and technological viewpoint, since there is a wide range of natural phenomena that can be modeled through fluid theory. Some common engineering examples are pumps, fans, turbines, airplanes, ships, rivers, windmills, pipes, and more recently, the hemodynamics of the arterial system. For computer graphics, the motivation for such interest relies in the potential applications of these methods for visual effects and in the beauty of the natural phenomena that are involved [1], [2]. In particular, techniques in the field of computational fluid dynamics (CFD) have been applied for fluid animation in applications such as virtual surgery simulators [3], visual effects [4] and games [5].

The traditional fluid animation methods rely on a top-down viewpoint that uses 2D/3D mesh-based approaches motivated by the Eulerian methods of Finite Element and Finite Difference in conjunction with Navier-Stokes equations for fluids [6], [7]. Alternatively, lattice methods comprised of the Lattice Gas Cellular Automata (LGCA) and Lattice Boltzmann (LBM) can be used. The basic idea behind these methods is that the macroscopic dynamics of a fluid are the result of the collective behavior of many microscopic particles. The LGCA simplifies the dynamics through simple and local rules for particle interaction and displacements. On the other hand, the LBM constructs a simplified kinetic model, a simplification of the Boltzmann equation, which incorporates the essential microscopic physics so that the macroscopic averaged properties obey the desired equations [8]. The LBM method has developed into an alternative and promising numerical scheme for simulating fluid flows in applications involving interfacial dynamics and complex boundaries.

Once performed the fluid simulation, scientific visualization methods can be applied in order to identify patterns of scientific interest inside the fields. Scientific Visualization is a computer-based field concerned with techniques that allow scientists to create graphical representations from the results of their computations, as well as to visualize features of interest in a data set obtained through imaging instruments [9].

Despite being computational expensive, visualization techniques have become essential in interpreting data because it exploits the dominance of the human visual channel (more than 50 percent of our neurons are devoted to vision) that makes humans experts at using their highly developed pattern-recognition skills to look anomalies. That is way visualization techniques are so important for scientific data analysis.

Contributions: In this work, we developed a computational system that incorporates a 3D LBM simulator and scientific visualization capabilities to analyze the generated fields. Specifically, we implemented the D3Q19 LBM model [8] with application for the simulation of a lid-driven cavity, a known benchmark in computational fluid dynamics [10]. The obtained fields (density and velocity) can be visualized using traditional techniques (color map and oriented arrows), available in the OpenGL library. Besides, GLUI user interfaces were designed for parameters choice, fluid domain setup and simulation control. Other visualization capabilities will be incorporated in the system soon. The development of this software is the contribution of this work.

The article is organized as follows. Section II reviews related works. Section III describes the Lattice Boltzmann technique and explain the D3Q19 LBM model. Section IV presents the software developed. In section V we present the simulation tests. Finally, section VI we offer final comments and future works.

II. Related Works

This work focuses on the simulation and visualization of three-dimensional fluids. The former involves numerous works that can be coarsely classified into nonphysically and physically-based models [1], [2]. Our work belongs to the latter class, which can be subdivided into PDEs and lattice-based techniques [11], [2]. PDE methods include continuous fluid equations, like Navier-Stokes, and numerical techniques based on discretization approaches, like Smoothed Particle Hydrodynamics (SPH) [12], method of characteristics [13], Moving-Particle Semi-Implicit methods [14], or Finite Element ones [6].

Lattice-based techniques, like HPP, FHP, and Lattice Boltzmann, work from a different viewpoint [11], [15]. For instance, the LBM method is based on the fundamental idea of constructing simplified kinetic models that incorporate the essential physics of microscopic processes so that the macroscopic averaged properties satisfy macroscopic equations. The LBM is especially useful for modeling complicated boundary conditions and multiphase interfaces [8]. Recent extensions of this method are described, including simulations of fluid turbulence, suspension flows, and reaction diffusion systems [16].

Lattice models have a number of advantages over more traditional numerical methods, particularly when fluid mixing and phase transitions occur [17]. Simulation is always performed on a regular grid, and can be efficiently implemented on a massively parallel computer. Solid boundaries and multiple fluids can be introduced in a straightforward manner and the simulation is done efficiently, regardless of the complexity of the boundary or interface [18]. In the case of LGCA, there are no numerical stability issues because its evolution follows integer arithmetic. For LBM, numerical accuracy and stability depend on the Mach number (max-speed/speed of sound). The computational cost of the LGCAs is lower than that for LBM and PDE-based methods. However, system parametrization (e.g., viscosity) is difficult to do in LGCA models, and the obtained dynamics is less realistic than for PDE-based models and LBM.

Real-time is a fundamental requirement for some applications, like games. So, in this case, the trade-off between realism and frame rate becomes the main challenge. Therefore, some authors proposed Navier-Stokes equation solvers based on finite difference approaches, with special care regarding stability and speed [19], as well as Lattice Boltzmann methods (LBM) for graphics hardware [20].

Finally, scientific visualization techniques must be applied to identify patterns of scientific interest inside the fields. The techniques in scientific visualization can be classified according to the data type they manage: scalar fields ($F : D \subset \Re^3 \rightarrow \Re$), vector fields ($F(x)$ is a vector, $x \in D \subset \Re^3$) and tensor fields compose the usual range of data types in this field. Henceforth, we have methods for scalar fields visualization (isosurface generation and volume rendering, colormap, among others), vector fields visualization (field lines generation, particle tracing, topology of vector fields, LIC, among others) and techniques for tensor fields (topology and hyperstreamlines) [9]. In this paper we have applied the traditional methods of color map, for density field visualization and oriented arrows for vector field visualization. Particle tracing and streamlines will be implemented soon.

III. The Lattice Boltzmann Method

In recent years, Lattice Boltzmann Methods (LBM) have taken the attention of the scientific community, due to their ease of implementation, extensibility and computational efficiency. Specifically in computational fluid dynamic, LBM has been applied due to its ease implementation of boundary conditions and numerical stability in wide variety of flow conditions with various Reynolds numbers [21]. The LBM has evolved from the Lattice Gas Cellular Automata (LGCA), which, despite its advantages, has certain limitations related to their discrete nature: the rise of noise, which makes necessary the use of processes involving the calculation of average values and little flexibility to adjust the physical parameters and initial conditions [21].

The LBM was introduced by [22], where the authors showed the advantage of extending the boolean dynamics of cellular automata to work directly with real numbers representing probabilities of presence of particles. In the LBM, the domain of interest is discretized in a lattice and the fluid is considered as a collection of particles. These particles move in discrete time steps, with a velocity pointing along one of the directions of the lattice. Besides, particles collide with each other and physical quantities of interest associated with the lattice nodes are updated at each time step. The computation of each node depends on the properties of itself and the neighboring nodes at the previous time step [21], [8]. The dynamic of this method is governed by the Lattice Boltzmann equation:

$$f_i(\vec{x} + \Delta_x \vec{c}_i,\ t + \Delta_t) - f_i(\vec{x}, t) = \Omega_i(f), \tag{1}$$

with $i = 1, ..., z$ and where f_i is the particle distribution function, $\vec{x}$ is the lattice node, $\vec{c}_i$ is one of the lattice directions,

Δ_x is the lattice spacing, Δ_t is the time step, $\Omega_i(f)$ is the collision term, and z is the number of lattice directions.

In the work presented in [23], the authors proposed to linearize the collision term Ω_i around its local equilibrium solution:

$$\Omega_i(f) = -\frac{1}{\tau}\Big(f_i(\vec{x},t) - f_i^{eq}(\rho,\vec{u})\Big), \tag{2}$$

where τ is the relaxation time scale and f_i^{eq} is the equilibrium particles distribution that is dependent on the macroscopic density (ρ) and velocity ($\vec{u}$). The parameter τ is related to diffusive phenomena in the problem, in this case with the viscosity of the fluid [8]. The general equation of the equilibrium function is given by [21]:

$$f_i^{eq} \;=\; \rho\omega_i\left[1 \;+\; \frac{(\vec{c}_i\cdot\vec{u})}{c_s^2} \;+\; \frac{(\vec{c}_i\cdot\vec{u})^2}{2c_s^4} \;-\; \frac{(\vec{u}\cdot\vec{u})}{2c_s^2}\right], \tag{3}$$

where ω_i are weights and the c_s^2 is the lattice speed of sound, which is dependent on the lattice.

There are different LBM models for numerical solutions of various fluid flow scenarios, where each model has different lattice discretization. The LBM models are usually denoted as DxQy, where x and y corresponds to the number of dimensions and number of microscopic velocity directions ($\vec{c}_i$) respectively. In this work we implement a 3D LBM model known as D3Q19. This model has 18 possibilities of non-zero velocities, as shown in Fig. 2, given by:

$\vec{c}_0 = (0,0,0)$,

$\vec{c}_1 = (1,0,0)v$,	$\vec{c}_2 = (0,1,0)v$,	$\vec{c}_3 = (-1,0,0)v$,
$\vec{c}_4 = (0,-1,0)v$,	$\vec{c}_5 = (0,0,-1)v$,	$\vec{c}_6 = (0,0,1)v$,
$\vec{c}_7 = (1,1,0)v$,	$\vec{c}_8 = (-1,1,0)v$,	$\vec{c}_9 = (-1,-1,0)v$,
$\vec{c}_{10} = (1,-1,0)v$,	$\vec{c}_{11} = (1,0,-1)v$,	$\vec{c}_{12} = (0,1,-1)v$,
$\vec{c}_{13} = (-1,0,-1)v$,	$\vec{c}_{14} = (0,-1,-1)v$,	$\vec{c}_{15} = (1,0,1)v$,
$\vec{c}_{16} = (0,1,1)v$,	$\vec{c}_{17} = (-1,0,1)v$,	$\vec{c}_{18} = (0,-1,1)v$,

where v is the particle velocity related to each direction $\vec{c}_i$.

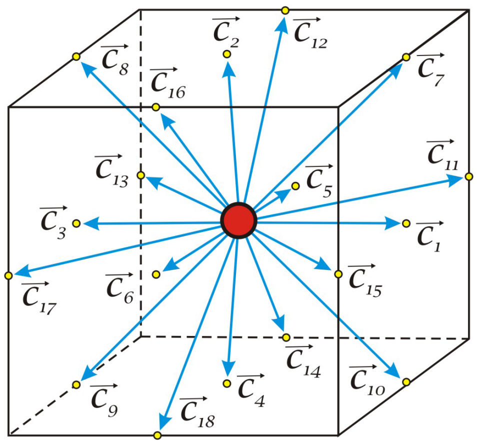


Fig. 2. The D3Q19 LB model, with 18 non-zero velocities.

The lattice speed of sound and the weights for the lattice of the D3Q19 LBM model are given by:

$$c_s^2 = \frac{1}{3},\quad \omega_0 = \frac{1}{3},\quad \omega_{1-6} = \frac{1}{18},\quad \omega_{7-18} = \frac{1}{36}, \tag{4}$$

where ω_0 is related to the rest particle. Replacing (4) in (3), gives us the equilibrium function for the D3Q19 LB model:

$$f_i^{eq} = \omega_i\left[\rho + 3\frac{(\vec{c}_i\cdot\vec{u})}{v^2} + \frac{9}{2}\frac{(\vec{c}_i\cdot\vec{u})^2}{v^4} - \frac{3}{2}\frac{(\vec{u}\cdot\vec{u})}{v^2}\right], \tag{5}$$

where $i = 0, ..., 18$.

Our interest relies on the macroscopic scale, where the physical macroscopic quantities seem to show a continuous behavior. Then, the macroscopic density (ρ) and velocity ($\vec{u}$) are calculated from the respective moments of the density distribution, as follows:

$$\rho(\vec{x},t) \;=\; \sum_{i=0}^{18} f_i(\vec{x},t), \tag{6}$$

$$\vec{u}(\vec{x},t) \;=\; \frac{1}{\rho(\vec{x},t)}\sum_{i=0}^{18} f_i(\vec{x},t)\vec{c}_i. \tag{7}$$

The main steps of the simulation algorithm can be summarized as follows:

Algorithm I: D3Q19 LBM Model

```
1) Initialization (t = 0):
    a) $\rho(\vec{x},0) = 1.0$
    b) $\vec{u}(\vec{x},0) = (0,0,0)$
    c) $f_i(\vec{x},0) = f_i^{eq}(\rho,\vec{u})$
2) Main Loop ($t = 1$ to $t_{max}$):
    a) Stream Step (8)
    b) Update macroscopic density (6) and velocity (7)
    c) Collision Step (9)
```

The Lattice Boltzmann equation (1) contains the two steps of the simulation, namely: stream and collision. These two steps can be computed separately, through the following expressions:

$$f_i^*(\vec{x},\ t+\Delta_t) = f_i(\vec{x}-\Delta_x\vec{c}_i, t), \tag{8}$$

and

$$f_i(\vec{x}, t+\Delta_t) = (1-\tau)f_i^*(\vec{x},\ t+\Delta_t) + \tau f_i^{eq}(\rho,\vec{u}). \tag{9}$$

Boundary conditions are fundamental features for fluid simulation. The standard boundary conditions for LBM simulations are no-slip walls [10]. It means that close to the boundary the fluid does not move at all. Hence, each cell next to a boundary should have the same amount of particles moving into the boundary as moving into the opposite direction. This will result in a zero velocity, and can be imagined as reflecting the particle distribution functions at the boundary. The reflection process is shown in Fig. 3. For the implementation, boundary and fluid cells need to be distinguished. Thus, in the streaming step, the previous algorithm must check the type of the cells: if the neighboring cell is a boundary, the opposite distribution function from the current cell would be taken, instead of applying the standard stream calculation (8).

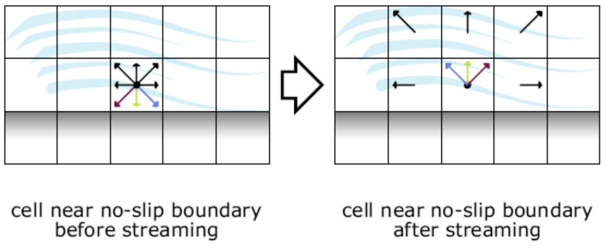


Fig. 3. No-slip obstacle cells directly reflect the incoming distribution functions. (*Source:* [10])

External forces can be added to the LBM model [10]. To introduce these, a force can be applied to all fluid cells through the following expression:

$$\vec{u} = \vec{u} + \tau\vec{F}, \tag{10}$$

where $\vec{u}$ is the macroscopic velocity, τ is the relaxation time scale and $\vec{F}$ is the external force. For the implementation, the previous algorithm must update the macroscopic velocity given by (7) adding the external forces calculated by (10). Thus, one may add a constant force such as gravity acting on the particles.

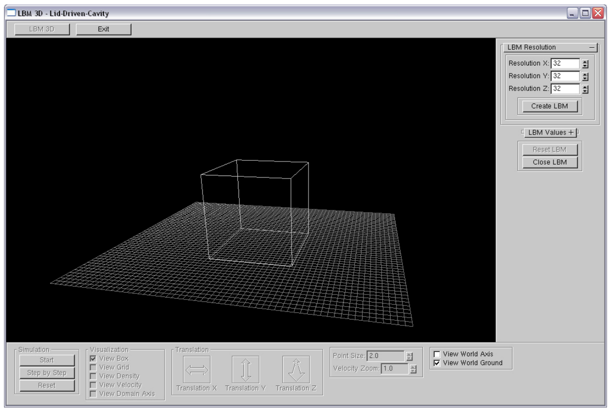
IV. Computational System

The software developed incorporates the D3Q19 LBM model for fluid simulation and scientific visualization techniques. The implementation follows the steps of Algorithm I, described in section III. For the LBM, initial conditions are usually specified in terms of macroscopic variables such as density and velocity. These macroscopic variables are translated into the corresponding microscopic particle distribution values for each node of the lattice, which is done by solving the equilibrium equation. Hence, the initial values for the density and velocity of each node are plugged into equation (5) and the equilibrium distribution values are set as the initial particle distribution values for each node.

The LBM simulation generates the density (scalar) and the velocity (vector) fields. The visualization of the density is performed through a color map for each cell of the lattice: we map the density range into a color scale and draw a single point. Therefore, the field is rendered as a color field in the display. The velocity field is visualized through oriented arrows for each cell, which size is scaled according to the velocity field intensity. Besides, the latter can be visualized using a color map, likewise performed for the density field.

The computational system was developed in $C/C++$ language, using the object oriented paradigm. The visualization was implemented in OpenGL [24], [25], a traditional library for computer graphics. The GLUT [26] programming interface was used to implement the interactions between OpenGL and the window system of our application. With GLUT we can use the graphical resources of OpenGL and access the window system functionalities of both Unix and Windows operating systems. The graphical user interface, composed by buttons, checkboxes, spinners, among others controls, is generated using the GLUI [27], [28] API, which is a GLUT-based $C++$ user interface library also portable for Unix and Windows systems. Therefore, the computational system developed can be executed in any platform running OpenGL.

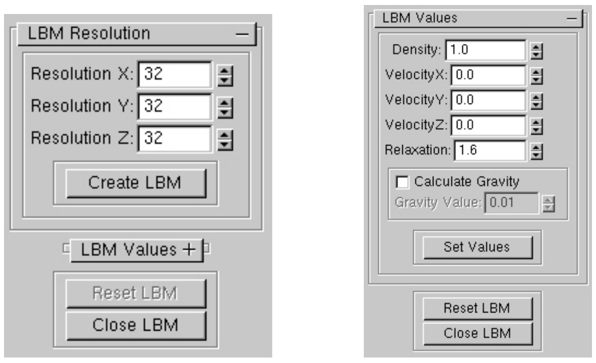
When using the application, the first step is to define the fluid domain otherwise the interface controls will not be available. The Fig. 4 shows the main window of the application with a panel, in the right-hand side, that groups text boxes for inputting integer values to set up the domain dimensions and start the LBM simulation.



Resolution X: 32
Resolution Y: 32
Resolution Z: 32
[✓ View Box]
View Grid
View Density
View Velocity
View Domain Axis
Point Size: 2.0
Velocity Zoom: 1.0
View World Axis
[✓ View World Ground]

Fig. 4. Main window of the computational system.

The Fig. 5.(a) shows details of the controls for defining domain dimensions and the LBM initialization. Once the domain dimensions are defined, the user just press the button *Create LBM* in order to initialize the simulation. Therefore, a D3Q19 model is instantiated with constant density and velocity fields (1.0 and $(0, 0, 0)$, respectively, for any lattice node). Besides, the relaxation time scale is set to 1.6.

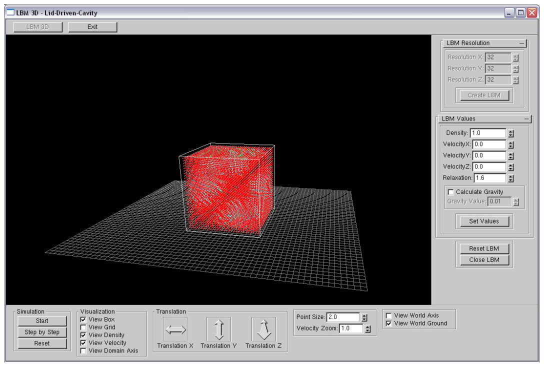


Resolution X: 32
Resolution Y: 32
Resolution Z: 32

Density: 1.0
VelocityX: 0.0
VelocityY: 0.0
VelocityZ: 0.0
Relaxation: 1.6
Calculate Gravity
Gravity Value: 0.01

(a) (b)

Fig. 5. (a)Menu to define the dimensions of the fluid domain and LBM instantiation. (b) Control panel for initial fields and relaxation time scale redefinition.



Resolution X: 32
Resolution Y: 32
Resolution Z: 32
Density: 1.0
VelocityX: 0.0
VelocityY: 0.0
VelocityZ: 0.0
Relaxation: 1.6
☐ Calculate Gravity
Gravity Value: 0.01
☑ View Box
☐ View Grid
☑ View Density
☑ View Velocity
☐ View Domain Axis
Point Size: 2.0
Velocity Zoom: 1.0
☐ View World Axis
☑ View World Ground

Fig. 6. Main window with density field visualization.

Then, the system displays a visualization of the density field, which is pictured on Fig. 6. Also, a new panel grouping text boxes for changing the initial fields and the relaxation time scale as well as to define the gravity intensity, becomes available, as shown in Fig. 5.(b).

Bellow the visualization area in the main window, there is a panel with text boxes to set visualization parameters, such as: arrow size, density point size, domain position, among others. Through the mouse the user can change the camera position for change the viewpoint, using transformations such as translation, rotation or zoom effects. Besides, it is possible to control the simulation by using the buttons:

- *Start/Pause*: This button starts and pauses the actual LBM instance.
- *Step by Step*: Advances the simulation one step further when pressed.
- *Reset*: Restart the simulation with the last initial conditions defined.

V. Results

In this section we describe some experiments using the LBM model of section III for fluid simulation and the software described in section IV for visualization of the velocity and density fields. We show the behavior of the fluids in two different set ups and highlight relevant aspects with the visualization. As explained in section IV, macroscopic view of the fields is via simple features such as colored points and arrows. However, such 3D visualization is uncomfortable and difficult to see. Thus, for the results above we chose to generate 2D images of a defined section of the lattice. We are visualizing the velocity field intensity using a color map that goes from the black (smaller velocities) to red (velocity intensity equal to one).

First experiment: the first example is the lid-driven cavity, a benchmark test for two-dimensional fluid solvers, but can be directly transferred to 3D. It consists of a rectangular domain, like the one presented on Fig. 4, filled with fluid and no-slip boundary conditions at the walls of the domain boundary. The top wall is moving with a constant velocity which transfers momentum for the fluid particles nearby the upper wall. After some simulation steps, a vortex in the middle of the domain is visible as well as smaller vortices in the corners which rotate in the opposite direction of the center vortex [10]. We initialize all cells with the equilibrium function (5), using $\rho = 1$ and $\vec{u} = (0, 0, 0)$. Following the reference [10], in the implementation of the lid-driven cavity we add to the fluid cells with $y = (SIZE - 2)$ a constant velocity of $(0.01, 0, 0)T$ during all simulation, where $SIZE$ is the y resolution. These lattice cells are called accelerator cells and, once their density does not change, we do not have mass conservation, which is a disadvantage of this implementation. The Fig. 7 shows some snapshots of the lid-driven cavity simulation with the following setup: domain dimensions 128×128, $\tau = 1.9$, $\rho = 1.0$, initial velocity far from the upper wall is null, accelerator cells velocity $(0.1, 0.0, 0.0)$. As expected, after some simulation steps (6700, in this case), we do observe a vortex in the middle of the domain as well as the smaller vortices in the corners of the fluid domain. The visual patterns observed agree with the results presented elsewhere [10].

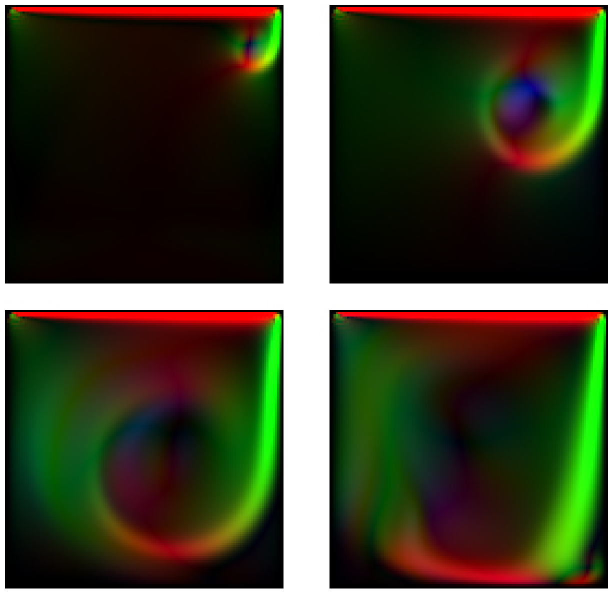


Fig. 7. Lid-Driven cavity results for time steps 700, 2700, 6700 and 10700.

Second experiment: the next experiment shows another example using a discontinuous density field. We define the fluid domain dimensions as 128×128 and an inner region with dimensions 32×32 located at the center of the fluid. Thus, we set to the inner lattice cells the initial density value 2.0 and for the other cells 1.0 as the initial density value. The other parameters are $\tau = 1.9$ and $\vec{u} = (0, 0, 0)$ for all cells. In a discontinuous density field, the region of higher density tends to spread to the region of lower density. As we can see in Fig. 8, the first two images show the propagation of the fluid before colliding with the boundary. From the third image we can see the result after the collision with the boundary, and as the simulation evolves, we realize that the velocity field tends to an equilibrium (softer colors), as seen in the last two images. Besides, we can observe that the simulation creates a velocity field with symmetric patterns. This happens because of the geometry of the lattice that tends to create major streams according to their directions.

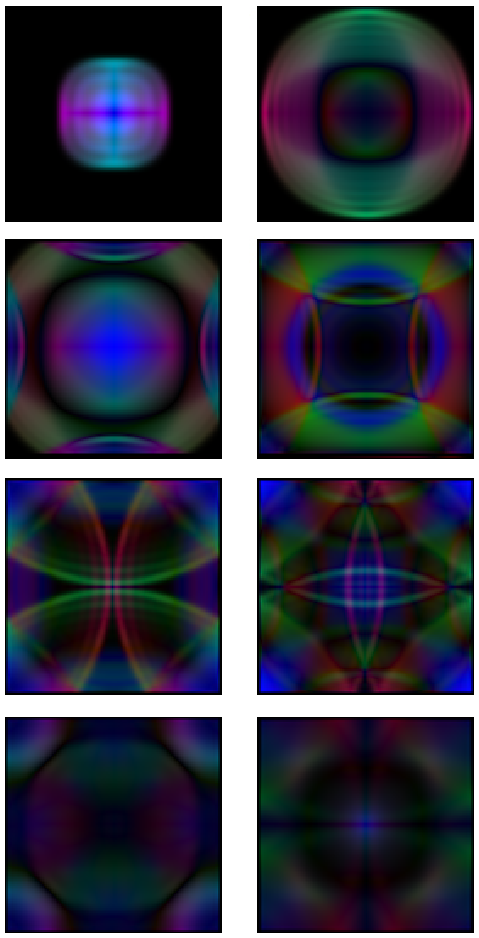


Fig. 8. Density gradient results for time steps 40, 80, 100, 140, 180, 200, 980 and 1280.

VI. Conclusions and Future Works

In this work we describe a 3D fluid simulator based on the D3Q19 LBM model. This simulator is integrated into a computer system with GLUI user interfaces and traditional fluid visualization techniques implemented through OpenGL facilities. Two fluid configurations was simulated and generated fields visualized using color map for scalar fields and arrows for velocity field. The visual patterns observed agree with the qualitative behavior expected. Further directions for this work will be to quantify the precision of our simulator. This can be done by comparing the simulation result with an analytic solution of Navier-Stokes equations. Besides, other visualization techniques must be incorporated in the software using the functionalities of VTK library.

References

[1] O. Deusen, D. S. Ebert, R. Fedkiw, F. K. Musgrave, P. Prusinkiewicz, D. Roble, J. Stam, and J. Tessendorf, "The elements of nature: interactive and realistic techniques," in *SIGGRAPH Course Notes*. ACM, 2004, p. 32.

[2] A. Iglesias, "Computer graphics for water modeling and rendering: a survey," *Future Gener. Comput. Syst.*, 2004.

[3] M. Müller, S. Schirm, and M. Teschner, "Interactive blood simulation for virtual surgery based on smoothed particle hydrodynamics," *Technol. Health Care*, vol. 12, no. 1, pp. 25–31, 2004.

[4] P. Witting, "Computational fluid dynamics in a traditional animation enviroment," in *SIGGRAPH*. ACM, 1999, pp. 129–136.

[5] M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa, "Point based animation of elastic, plastic and melting objects," in *ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 2004, pp. 141–151.

[6] N. Foster and D. Metaxas, "Modeling the motion of a hot, turbulent gas," in *SIGGRAPH*. ACM, 1997, pp. 181–188.

[7] J. Stam, "Flows on surfaces of arbitrary topology," in *SIGGRAPH*. ACM, 2003, pp. 724–731.

[8] S. Chen and G. D. Doolen, "Lattice boltzmann method for fluid flows," *Annual Review of Fluid Mechanics*, vol. 30, pp. 329–364, 1998.

[9] R. E. J. L. H. H. K. A. K. S. N. G. P. F. T. D. ROSENBLUM, L.; EARNSHAW, *Scientific visualization - advances and challenges*. New York Academic Press, 1994.

[10] N. Thürey, "A Lattice Boltzmann method for single-phase free surface flows in 3D," Master's thesis, Dept. of Computer Science 10, University of Erlangen-Nuremberg, 2003.

[11] U. Frisch, D. D'Humières, B. Hasslacher, P. Lallemand, Y. Pomeau, and J.-P. Rivet, "Lattice gas hydrodynamics in two and three dimension," *Complex Systems*, vol. 1, pp. 649–707, 1987.

[12] G. R. Liu and M. B. Liu, *Smoothed particle hydrodynamics : a meshfree particle method*. World Scientific, 2003.

[13] J. Stam, "Stable fluids," in *SIGGRAPH*. ACM, 1999, pp. 121–128.

[14] S. Premoze, T. Tasdizen, J. Bigler, A. Lefohn, and R. T. Whitaker, "Particle-based simulation of fluids," 2003.

[15] R. Benzi, S. Succi, and M. Vergassola, "The lattice boltzmann equation: theory and applications," *Physics Reports*, vol. 222, no. 3, pp. 145 – 197, 1992.

[16] X. Wei, S. Member, W. Li, K. Mueller, and A. E. Kaufman, "The lattice-boltzmann method for simulating gaseous phenomena," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, pp. 164–176, 2004.

[17] D. H. Rothman and S. Zaleski, "Lattice-gas models of phase separation: Interface, phase transition and multiphase flows," *Rev. Mod. Phys*, vol. 66, pp. 1417–1479, 1994.

[18] C. A. G. J. M. Buick, W. J. Easson, "Numerical simulation of internal gravity waves using a lattice gas model," *International Journal for Numerical Methods in Fluids*, vol. 26, no. 6, pp. 657–676, 1998.

[19] J. Stam, "Real-time fluid dynamics for games," in *In Proceedings of the Game Developer Conference*, 2003.

[20] Q. F. F. Z. Zhao, Y. and A. E. Kaufman, "Flow simulation with locally-refined lbm," in *Proceedings of the Symposium on Interactive 3D Graphics and Games*. ACM, 2007, pp. 181–188.

[21] B. Chopard, P. Luthi, and A. Masselot, "Cellular automata and lattice boltzmann techniques: An approach to model and simulate complex systems," in *Advances in Physics*, 1998.

[22] G. R. McNamara and G. Zanetti, "Use of the boltzmann equation to simulate lattice-gas automata," *Phys. Rev. Lett.*, vol. 61, no. 20, pp. 2332–2335, November 1988.

[23] F. J. Higuera, J. Jimenez, and S. Succi, "Boltzmann approach to lattice gas simulations," *Europhys. Lett.*, vol. 9, 1989.

[24] D. Shreiner, *OpenGL Rerefence Manual: The Official Reference Document to OpenGL*, 4th ed. http://www.opengl.org/: Addison-Wesley Professional, 2004.

[25] R. S. Wright, B. Lipchak, and N. Haemel, *OpenGL SuperBible (4rd Edition)*. Addison-Wesley Professional, 2007.

[26] M. J. Kilgard, *The OpenGL Utility Toolkit (GLUT): Programming Interface*, november 1996.

[27] P. Rademacher, *GLUI: A GLUT-Based User Interface Library*, June 1999.

[28] N. Stewart, *GLUI User Interface Library*, http://glui.sourceforge.net, 2006.