A lattice Boltzmann implementation of the reference map technique

a thesis presented
by
Yue Sun
to
Institute of Applied Computational Science,
School of Engineering and Applied Sciences

in partial fulfillment of the requirements
for the degree of
Master of Engineering
in the subject of
Computational Science and Engineering

Harvard University
Cambridge, Massachusetts
April 2020

Thesis advisor: Professor Christopher H. Rycroft Yue Sun

A lattice Boltzmann implementation of the reference map technique

Abstract

This thesis presents a hybrid fully-Eulerian numerical method for fluid–structure interaction (FSI) simulation. The new method (LB-RMT) combines the reference map technique (RMT) and the lattice Boltzmann (LB) method to simulate the interaction between the solid and the fluid, and incorporates an extrapolation zone to update the solid-fluid interface. It simplifies the inter-phase coupling by taking advantage of the two Eulerian-based methods: the RMT permits both the solid and the fluid to be represented on a single fixed mesh, and the LB method uses a mesoscopic probabilistic view of particle motion to simplify macroscopic hydrodynamics update. The LB-RMT also provides a novel approach to mesoscopically simulating solid deformation. Examples consisting of a quasi-incompressible Navier–Stokes fluid with compressible neo-Hookean solids show this method is flexible and robust in handling standard fluid–structure interactions and deformable solids with sharp corners bending and twisting in fluid. A convergence study demonstrates the LB-RMT is of second-order accuracy, improved from the previous first-order RMT. The method can be extended to handle dissolving solids and multispecies cases like fluid–solid–gas simulation in two and three dimensions.

Contents

0 Introduction 1

1 Theory of Solid Deformation and Lattice Boltzmann Fluid 8
- 1.1 Reference Map Technique 9
- 1.2 Lattice Boltzmann Method 12
- 1.3 Fluid–Structure Interaction 19

2 Lattice-Boltzmann-Based Reference Map Technique 22
- 2.1 Lattice-Boltzmann-Based Solid Deformation 23
- 2.2 Modification of the Lattice Boltzmann Method with Forces 25
- 2.3 Overview of the Algorithm 27
- 2.4 Numerical Method 27

3 Results of the Lattice-Boltzmann-Based Reference Map Technique 40
- 3.1 Convergence Study of Plane Shear Waves in Solid Deformation 41
- 3.2 Deformable Sheet in the Poiseuille Flow 45
- 3.3 Rotors in Initially Quiescent Flow 51

4 Conclusion 58

Appendix A Units Conversion 61

References 65

Listing of figures

1 Illustrations of the domain discretization for the four types of numerical methods in fluid–structure interactions. (i) Mesh-free approach: the fluid and the solid are represented by particles. (ii) Lagrangian-based method for both phases: unstructured meshes are initialized to discretize the solid geometry and the submerging fluid. (iii) Eulerian-based method for the fluid and Lagrangian-based method for the solid: the fluid is discretized by a fixed mesh, and the solid is represented by markers. These solid markers can be off grid points. (iv) Eulerian-based method for both phases: the fluid and the solid are discretized by one fixed mesh, and the solid boundary is represented by a level set function. 4

1.1 Illustration of the hyperelastic solid deformation, where a time-dependent mapping $\vec{\chi}(\vec{X}, t)$ is applied to an initially undeformed solid with reference coordinate system $\vec{X}$ and results in a deformed coordinate system at time t. 9

1.2 Diagram of the D_2Q_9 lattice model for the two-dimensional LB method. Each f_i is a probability distribution function of a particle velocity at (i, j) in the direction of the black arrow. The grey lattices are the neighboring lattices, which will receive the f_i's contribution in the next timestep. 13

1.3 Diagram of the streaming routine for the two-dimensional LB method. The grey arrows and labels show the original position of the nine post-collision populations $\widehat{f_i}$. Each $\widehat{f_i}$ moves from its grey position in the direction of dashed arrow to its neighboring lattice. $\widehat{f_i}$ then becomes the new f_i of the neighboring lattices in the next timestep. 17

1.4 Illustration of the reference map configuration for a fluid–structure interaction with a deformable solid immersed in fluid on a fixed Eulerian grid. A signed distance function denotes the level set function $\phi(\vec{x}, t)$, where a positive distance represents the fluid and a negative represents the solid. The extrapolation zone generates a smooth transition between the two phases, defined as $0 \leq \phi \leq \epsilon$ where ϵ is the width of the extrapolation zone. 21

2.1 Grid setup of the lattice-Boltzmann-based reference map technique for a two-dimensional fluid–structure interaction simulation. The simulation domain has length L and height H, with $n_x \times n_y$ physical nodes. Two layers of buffer nodes are padded to each side, giving a total of $(n_x + 4) \times (n_y + 4)$ nodes in the grid. Solid nodes are represented in red, and fluid nodes are in blue. The solid–fluid interface is marked by the purple line, and the extrapolation zone between the two phases is colored in light blue. There are five layers in the extrapolation zone based on their distances to the interface. The nodes in each layer are assigned with a corresponding indicator value, *e.g.* nodes in the first layer are given an indicator of value 1. 29
2.2 Illustration of the reference map extrapolation in the extrapolation zone. The extrapolation starts from the first layer in the extrapolation zone then moves outward to the next layer after all nodes have been extrapolated. A scan window of width $r = 5$ centered at the target node is initialized, and the extrapolated reference map is calculated using the solid nodes inside the scan window. If the result is invalid, then increase the scan window width r to include more solid nodes. 31
2.3 Stencil setup for the calculation the divergence of solid stress. The red circle represents the current solid node, and the red crosses represent the mid-grid intermediate solid stresses. To compute these stresses, for example the left solid stress $(\sigma_s)_{i-\frac{1}{2},j}$, six solid nodes in total are involved in the gradient calculations: (i, j), $(i - 1, j)$ for the x-direction derivatives, and $(i, j + 1)$, $(i - 1, j + 1)$, $(i, j - 1)$, $(i - 1, j - 1)$ for the y-direction derivatives. 33
2.4 Diagram of the periodic boundary conditions on a simplified simulation domain. Along the x-axis, at the right fluid boundary, the right-outward populations (f_1, f_5, f_8) are copied to the first layer in the left buffer. And at the left fluid boundary, the left-outward populations (f_3, f_6, f_7) are also copied to the first layer in the right buffer. The buffered values in the first layer will then be streamed back to the fluid boundaries, thus imposing periodicity. Along the y-axis, the operations are analogous. At the top fluid boundary, the up-outward populations (f_2, f_5, f_6) are copied to the first layer in the bottom buffer. And at the bottom fluid boundary, the down-outward populations (f_4, f_7, f_8) are also copied to the first layer in the top buffer. 37

2.5 Diagram of the no-slip boundary conditions on a simplified simulation domain. For one node on the top first buffer layer, the stored f_4, f_7, f_8 populations correspond to the f_2 of its lower, f_5 of its lower-left, and f_6 of the lower-right fluid populations. For one node on the bottom first buffer layer, the stored f_2, f_5, f_6 populations correspond to the f_4 of its upper, f_7 of its upper-right, and f_8 of the upper-left fluid populations. For one node on the left first buffer layer, the stored f_1, f_5, f_8 populations correspond to the f_3 of its right, f_7 of its upper-right, and f_6 of the lower-right fluid populations. For one node on the right first buffer layer, the stored f_3, f_6, f_7 populations correspond to the f_1 of its left, f_8 of its upper-left, and f_5 of the lower-left fluid populations. The four corners in the first buffer layer are handled differently, each with only one population bouncing back to the fluid region. 38

3.1 log–log plot of the grid spacing Δx and the simulation error $E_{\vec{\xi}}$, E_v. 43
3.2 Plot of the relation between the relaxation time τ_{LB} and the simulation error $E_{\vec{\xi}}$, E_v. Modified simulation errors are plotted in dashed lines. Though the simulations are more accurate with the modification in the input viscosity, the log–log plot of the modified simulation errors does not conclude more information about their convergence. 44
3.3 Snapshots at $t = 0, 5000\Delta t, 10000\Delta t$ of the density fields of the channel flow and the deformable sheet. Density undergoes very little change in the first half of the simulation, and is conserved in most regions. 47
3.3 Snapshots at $t = 15000\Delta t, 20000\Delta t, 24500\Delta t$ of the density fields of the channel flow and the deformable sheet. Density changes are more prominent near the solid boundary when the solid is more deformed. 48
3.4 Snapshots at $t = 0, 5000\Delta t, 10000\Delta t$ of the velocity field. A parabolic velocity profile of the Poiseuille flow develops along the channel. The fluid moves the sheet forward. 49
3.4 Snapshots at $t = 15000\Delta t, 20000\Delta t, 24500\Delta t$ of the velocity field. The stretched sheet has lower velocity inside the solid, and changes the standard Poiseuille flow velocity profile near the no-slip wall boundaries. 50
3.5 Snapshots at $t = 400\Delta t, 600\Delta t, 800\Delta t$ of the vorticity field of one rotor. Rotor spins from clockwise to counter-clockwise. Vortices are shed from the solid–fluid interface, and the fluid near the rotor has been excited. 52
3.5 Snapshots at $t = 1000\Delta t, 1200\Delta t, 1400\Delta t$ of the vorticity field of one rotor. The rotor continues to spin counter-clockwise, and is very twisted at $t = 1400\Delta t$. More fluid has been excited due to the rotor rotation. 53
3.6 Snapshots at $t = 2000\Delta t, 2200\Delta t, 2400\Delta t$ of the vorticity field of two rotors. The left rotor spins clockwise and the right counter-clockwise. Opposite vortices are shed from the seven points of the rotors. 55

3.6 Snapshots at $t = 2600\Delta t, 2800\Delta t, 3000\Delta t$ of the vorticity field of two rotors. The left rotor switches spinning directions to counter-clockwise and the right to clockwise. The entire channel flow has been excited. 56
3.7 Snapshots at $t = 9600\Delta t, 24300\Delta t, 40000\Delta t$ of the vorticity field of two rotors. The left rotor experiences more distortion due to flow disturbance, where some of its angles have swimmer-like movements. 57

To my parents,
Shilin Sun and Li Sun.

Acknowledgments

This master's thesis would not have been possible without the support and encouragement from all the incredible people I have met at Harvard and all my family and friends back home. I want to first express my deepest gratitude to my advisor Professor Christopher H. Rycroft. I am sincerely thankful for all his guidance in the past two years, and especially during the last two months for his academic and moral support. I have learned so many invaluable things from him throughout this thesis journey, from how to conduct research, to how to format a thesis, to how to present a talk, even to how to pick a colormap. I have also learned from him the awesomeness of the rainbow colors, and the visual power they process in conveying physical information and aesthetic pleasure. He has helped me realize the enjoyment of doing research, which has strengthened my determination to continue my graduate study. I am very lucky and looking forward to having him as my Ph.D. advisor in the future years.

Then I want to express my gratitude to my committee members, Dr. David Sondak and Professor Efthimios Kaxiras, for their insightful feedback on my thesis draft and their encouraging guidance in the past year. I am especially thankful for Dr. Sondak letting me be a teaching fellow for the undergraduate fluid dynamics course this spring. This teaching experience has refreshed and deepened my understanding of fluids and allowed me to share my love of fluid dynamics to more students.

I also want to extend my special thanks to Professor Sauro Succi for his expert advice on the lattice Boltzmann method. In addition, I am grateful for his patience and kindness with me when I was initially intimidated by the subject and dropped his course two years ago. He has helped me establish more confidence and a deeper understanding of the method the second time I took the course.

Throughout my two-year master's study, I have met many wonderful graduate students and staff in IACS. I am very grateful for all their support in my academic and non-academic life, and the sense of belonging. I have also been fortunate to be around the wonderful Rycroft Group. I have learned many things from everyone in the group, during journal club or random conversations, especially how they connect science and art in their work. Seeing the way they do research, as well as the charming group atmosphere, have inspired me to become a better graduate student. Additionally, I am very thankful for all the feedback and comments they gave me in my practice defense talk. In the past two months, seeing everyone in the Zoom group meeting has been my highlight of the week.

Lastly, I want to thank my parents, Shilin Sun and Li Sun, for their unconditional love, trust, and support. Though they have not fully understood my research topic or my passion for animation, they have always encouraged me to move forward and realize my dream. They have also tried to convince me that my research is very related to their field in civil engineering (and it is.) I also want to thank all my friends for their encouragement along the way, and for bearing with my anxiety every time I text or call. I especially want to thank Yuan Tian, Yiming Zhang, Yiyi Zhang, Chang Zhao, Jovana Andrejevic, Baptiste J. Lemaire, Nicholas Z. Stern and Sophie Li for their listening to me rambling about my study in the past two years and helping me practice my defense presentation.

This semester has been unpredictably unprecedented. I have been wanting to write something to document my feelings and thoughts upon this fluidic, anxious, and trying time since March 12, 2020, my first day of staying-at-home amid the COVID-19 global pandemic. Earlier this year, I had thought about locking myself in for a month to compile my thesis, but I never imagined everyone would be staying at home, and everything would be virtual-only. My thesis defense was given at the exact 50-day mark of me exclusively staying indoor. I had imagined my committee, my friends, and my parents gathering in the same room watching me present my work. Only this time, via a virtual Zoom room. As of now, all my feelings and thoughts seem vain and unimportant compared to the real world. All I can think about is not how fulfilled and relieved I am with completing the thesis, but how I genuinely hope things will be back to normal, and everyone I know, and their family, will be safe and well. I cannot wait the day to see everyone in person and resume the postponed celebration with them.

0 Introduction

Fluid–structure interaction (FSI), commonly pivoted around the study of the interaction between a moving or stationary deformable solid with its submerging or internal fluid (Dowell & Hall 2001; Wang 2008; Bazilevs et al. 2013), poses intriguing numerical challenges for computational fluid and solid mechanics. Since the FSI problems live in both the fluid and the solid worlds, the numerical methods chosen for such simulations need to be robust and flexible enough to handle both phases.

Typical numerical methods for fluid simulations use an Eulerian perspective (Chorin 1967; Anderson et al. 1997; Versteeg & Malalasekera 1995; Hirt et al. 1974), where often a rectangular or other standard-shaped simulation domain is discretized by a fixed mesh. Each grid point contains information on the hydrodynamic fields like velocity, density, and pressure at that particular position. An Eulerian-based method tracks the time-dependent changes in hydrodynamics of the entire fluid domain and offers three advantages: a direct global description of the hydrodynamic fields, a straightforward fixed mesh setup, and an easy implementation of the incompressible flow constraint if needed.

Whereas for solid simulations, or interface tracking between the fluid and the solid, Lagrangian-based methods are more favored (Zienkiewicz & Taylor 1967; Sulsky et al. 1994; Hoover et al. 2006; Belytschko et al. 2013) since the simulation only tracks a small mesh or a set of markers that represents either the solid or its boundary. Mesh points or markers move with the solid, replacing the need to track a fixed-shaped simulation domain that encloses the entire trajectory of the solid movement. Given the extra operations required to update the solid mesh or markers, a Lagrangian-based method is still more consistent with the solid framework and thus more often used to describe its dynamics.

The key challenge for FSI simulations is to reconcile the discrepancy in the preferred discretization frameworks for the fluid and the solid, and to design a method that can connect both phases. Existing methods have successfully addressed this concern and produced credible results in airfoil modeling (Shur et al. 1999; Shyy et al. 2010), arterial blood flow (Bazilevs et al. 2006; Gerbeau & Vidrascu 2003), and biolocomotion (Shelley & Zhang 2011; Zöttl & Stark 2012). These methods can be loosely categorized into four types. One type is to simulate both the fluid and the solid in a mesh-free approach, like the immersed particle method (Rabczuk et al. 2010), the material point method (Stomakhin et al. 2013) and the smoothed particle hydrodynamics method (Gingold & Monaghan 1977; Tasora et al. 2015). Without the need for a mesh, this type of method removes any potential complexity led by different choices of frameworks, and automatically ensures mass conservation and generates the interface between phases. It can additionally generate particle-based fluid, where a splash of water may be a serendipitous byproduct that is sometimes laborious to reproduce with Eulerian-based methods. Another type is to use Lagrangian-based methods for both the fluid and the solid, like the finite-element procedure (Rugonyi & Bathe 2001; Bathe 2006). In addition to the standard Lagrangian finite-element discretization of the solid, a second Lagrangian mesh is introduced to represent the fluid. This fluid mesh has to conform to the solid boundary, and is often unstructured to describe complex and flexible topologies (Persson et al. 2007, 2009; Persson & Peraire 2009).

The next type of FSI methods, also has the most applications, is to use Eulerian-based method for the fluid and Lagrangian-based method for the solid (Peskin 1977; Mittal & Iaccarino 2005). It discretizes the fluid on a fixed mesh, and uses an additional set of Lagrangian markers to represent the solid boundary. One technique to bridge the two phases is to calculate the surface tension forces along the solid boundary markers and then forward these forcing terms to update the fluid domain. This type of methods provides an intuitive discretization setup of the two phases, and is broadly used to track rigid or deformable solids in fluid flow. One widely cited example is the immersed boundary method (Peskin 2002; Griffith et al. 2009; Fai et al. 2013), which has long been used to simulate swimming, flapping, and elastic objects in biomechanical applications (Zhang et al. 2000; Watanabe et al. 2002; Zhu & Peskin 2002; Connell & Yue 2007).

These methods above more or less leverage the Lagrangian framework to describe the fluid and the solid, adding time-dependent complexity of tracking and updating the Lagrangian markers. Without the requisite of utilizing Lagrangian frameworks or switching between Eulerian and Lagrangian frameworks, the discretization and implementation of FSI methods will be only on one fixed grid, where both the fluid and solid can be described using Eulerian methods. The last type of FSI methods is an example of such, which either uses a level set function to describe the topological changes of the solid–fluid interface, like the level set method (Sethian 1999; Osher & Sethian 1988), or uses the usual Eulerian framework for the fluid and an Eulerian framework to describe the continuum solid (Udaykumar et al. 2003; Rycroft & Gibou 2012; Rycroft et al. 2015) of large-strain (Gurtin et al. 2010) and hyperelastic materials (Truesdell 1955), like the reference map technique (Kamrin & Nave 2009; Kamrin et al. 2012; Valkov et al. 2015; Rycroft et al. 2018). The latter example uses a reference map field, defined as an Eulerian mapping from the deformed framework to the unreformed framework of the solid on the fixed mesh, and a level set function to describe the interface between the fluid and the solid. This method leads to a unified, simple, explicit finite-difference calculation of solid deformation, feasible to be combined with any finite-difference methods for fluid simulation.

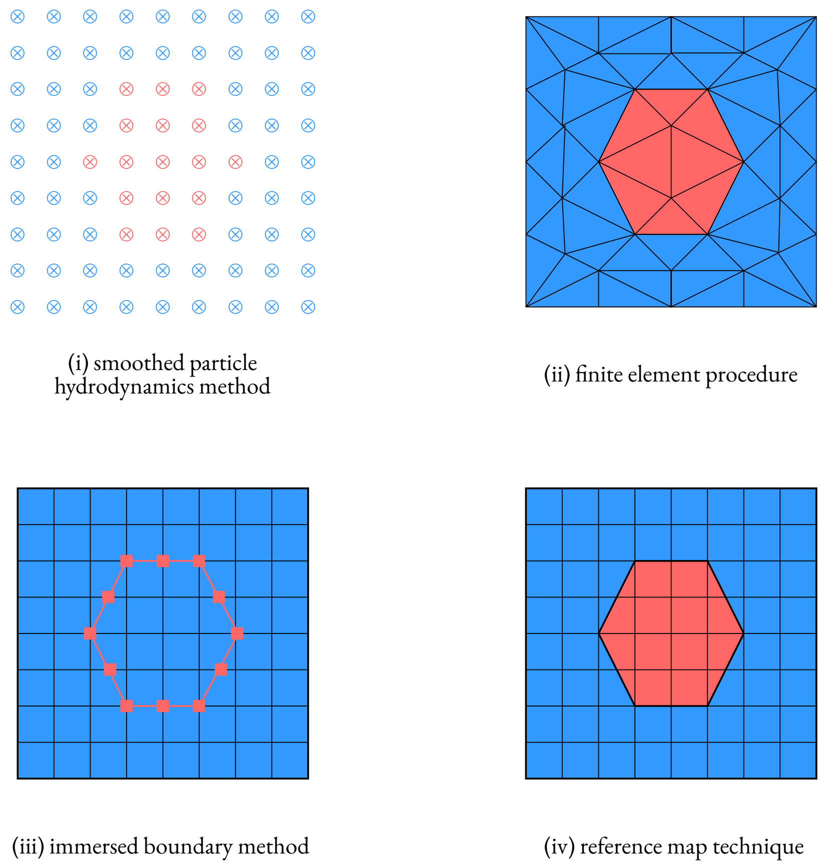
(i) smoothed particle hydrodynamics method
(ii) finite element procedure
(iii) immersed boundary method
(iv) reference map technique

Figure 1: Illustrations of the domain discretization for the four types of numerical methods in fluid–structure interactions. (i) Mesh-free approach: the fluid and the solid are represented by particles. (ii) Lagrangian-based method for both phases: unstructured meshes are initialized to discretize the solid geometry and the submerging fluid. (iii) Eulerian-based method for the fluid and Lagrangian-based method for the solid: the fluid is discretized by a fixed mesh, and the solid is represented by markers. These solid markers can be off grid points. (iv) Eulerian-based method for both phases: the fluid and the solid are discretized by one fixed mesh, and the solid boundary is represented by a level set function.

The reference map technique (RMT) plausibly inherits all the advantages of an Eulerian-based method, along with its fully-Eulerian framework in discretizing continuum solids. It has proven first-order accuracy and stability to simulate large solid deformation in fluid, including examples of flexible rods, a flapping swimmer, and a rotor with sharp corners in incompressible fluid (Rycroft et al. 2018). The current implementation of the RMT uses second-order finite-difference methods to calculate the advective terms and stresses. It uses the projection method of Chorin (1967, 1968) to impose incompressibility for the fluid, whereby a finite-element-based Poisson problem for the pressure is solved (Bell et al. 1989; Almgren et al. 1996; Yu et al. 2003, 2007), which is used to project the velocity to be divergence free. While the Poisson problem can be solved efficiently using the multigrid method (Briggs et al. 2000), it still represents the most computationally expensive part of the current RMT implementation, taking upward of two-thirds of the total computation time (Rycroft et al. 2018).

A promising alternative to the current finite-difference-based implementation is to use the lattice Boltzmann (LB) method for the fluid update (Succi 2001; Krüger et al. 2017; Chen & Doolen 1998). The LB method is an Eulerian-based simulation technique for computational fluid dynamics, which originated from the kinetic theory of gases (Higuera & Jiménez 1989; Rivet & Boon 2005; Succi 2001) in statistical physics. It also uses a fixed mesh to represent a rectangular simulation domain, where each grid point corresponds to a fluid node. Rather than containing the macroscopic hydrodynamic fields as the fluid nodes in the other Eulerian-based methods, the LB nodes are assigned with probability distribution functions. These probability distribution functions, also denoted as the particle populations (Krüger et al. 2017), are the statistical quantities associated with each fluid node to represent the particle density in the designated discrete velocity space at the current space and time. The macroscopic quantities in the continuity and Navier–Stokes equations (Batchelor 2000; Landau & Lifshitz 1987), like the density and velocity, are retrieved from this statistical view of fluid motion as moments of the populations (He & Luo 1997). The LB method has unique advantages in three distinct aspects: First, it requires no special adjustments to include complex rigid obstacles (Succi et al. 1989;

Martys & Chen 1996) and minimal additions to handle multiphase fluids (Aidun & Clausen 2010; He et al. 1999; Grunau et al. 1993), thus making the method popular in porous media (Pan et al. 2006; Guo & Zhao 2002; Sahimi 2011) and multicomponent flow (Shan & Doolen 1995; Martys & Chen 1996). Second, since the LB method is quasi-incompressible (Succi 2001; Krüger et al. 2017), *i.e.* it is incompressible if the simulated flow has been fully-developed and the simulation domain is a long channel to decrease back-propagating waves at the initial frames of simulation, the incompressibility constraint has been encoded into the method. No extra calculation or method is needed to impose this constraint. Lastly, the LB method is also suitable for parallelization (Kandhai et al. 1998; Amati et al. 1997; Bernaschi et al. 2009) due to the nature of its loop-based implementation. Heavy on the memory allocation, the method can be improved with the multi-threaded performance and be customized for diverse, complex obstacles and boundary conditions.

The other reason to link the LB method with the RMT is its analogy to the immersed-boundary-lattice-Boltzmann method (IB-LBM), a method (Feng & Michaelides 2004) that combines the IB-based solid boundary with the LB-based fluid for FSI simulations. Since the IB method uses the force density (Peskin 1977) along the Lagrangian solid markers to represent the boundary, it can be coupled with any fluid solver that supports external forcing, like the LB method (Guo et al. 2002; Huang et al. 2011; Ginzburg et al. 2008; Shan & Chen 1993; He et al. 1998). Though this method has desirable results in simulating elastic objects in viscous flow like flexible sheets (Zhu et al. 2011) and red blood cells (Bagchi 2007; Zhang et al. 2007; Sui et al. 2008; Li et al. 2013; Wang et al. 2013; Krüger et al. 2013), it still requires an additional Lagrangian framework to represent the solid boundary, and extra calculations to interpolate the velocity of the solid markers using the fluid nodes. Another limitation of IB-LBM lies in its simulation domain, where the entire mesh is filled with fluid nodes and only a set of mesh-independent markers to represent the solid boundary. This setup leaves no information to simulate the deformation or the kinematics inside the solid. One remedy is to add interior markers like the direct-forcing and fictitious-domain methods (Nie & Lin 2010; Feng & Michaelides 2009).

This thesis combines the RMT and the LB method to numerically simulate the interaction between deformable solids in submerging incompressible fluid. This hybrid fully-Eulerian method requires only one fixed mesh, where each grid point represents either a fluid node or a solid node, depending on the level set function that separates the fluid and the solid. The RMT operates on the solid nodes and describes the solid deformation by tracking the solid reference map and calculating the solid stress using the deformation gradient tensor (Plohr & Sharp 1988; Trangenstein & Colella 1991; Liu & Walkington 2001) and a chosen solid constitutive relation. This solid stress is then forwarded to the LB method as external force density on the solid nodes. The LB method updates all the nodes, with either standard LB routines for fluid nodes, or forcing-term modified LB routines for solid nodes. An additional extrapolation zone (Valkov et al. 2015; Rycroft et al. 2018) around the solid nodes is appended to extrapolate the solid reference map and update the solid and fluid nodes based on the level set function. This new method brings together the unique advantages of its two parent methods, *i.e.* representing both the fluid and the solid with one method on a fixed mesh, and calculating the hydrodynamics without heavy-lifting numerical burden. It also offers a novel perspective on simulating solid deformation and fluid–structure interaction in a mesoscopic probabilistic view.

The thesis is divided into four parts. Chapter 1 introduces the theory of the reference map technique for solid deformation and the lattice Boltzmann method for fluid dynamics, supplemented with the algorithm overviews of two methods and a short description of the fluid–structure interaction configuration. Chapter 2 studies the link between the two methods and how to derive the lattice-Boltzmann-based reference map technique (LB-RMT) for fluid–structure interaction simulations. Discretization details of the numerical methods are also provided in this chapter. Chapter 3 presents three examples: A convergence study of solid shear wave simulation, a deformable sheet in the Poiseuille flow, and rotors in initially quiescent flow. Chapter 4 concludes the strengths and weaknesses of the hybrid method and discusses its future extensions and applications.

1

Theory of Solid Deformation and Lattice Boltzmann Fluid

The proposed hybrid method for simulating fluid–structure interaction is based on the reference map technique (Rycroft et al. 2018) and the lattice Boltzmann method (Succi 2001). This chapter introduces the fundamentals of the reference map technique, which uses an Eulerian framework to describe solid deformation, and the lattice Boltzmann method, which mesoscopically reconstructs the macroscopic fluid quantities.

1.1 Reference Map Technique

1.1.1 Overview of the Reference Map Technique

The key to the reference map technique is to describe the solid deformation (Lubliner 2008; Gurtin et al. 2010) in the Eulerian framework. We first introduce an undeformed reference configuration of the hyperelastic solid at time $t = 0$ with coordinate system $\vec{X}$, and after some time, the reference configuration is deformed to a new coordinate system $\vec{x}$. Consider a time-dependent mapping $\vec{\chi}$ from the undeformed coordinate system $\vec{X}$ to the deformed coordinate system $\vec{x}$, *i.e.* $\vec{x} = \vec{\chi}(\vec{X}, t)$. The deformation gradient tensor $\mathbf{F}$ (Plohr & Sharp 1988; Trangenstein & Colella 1991; Liu & Walkington 2001) is denoted as

$$\mathbf{F}(\vec{X}, t) = \frac{\partial \vec{\chi}}{\partial \vec{X}}. \tag{1.1}$$

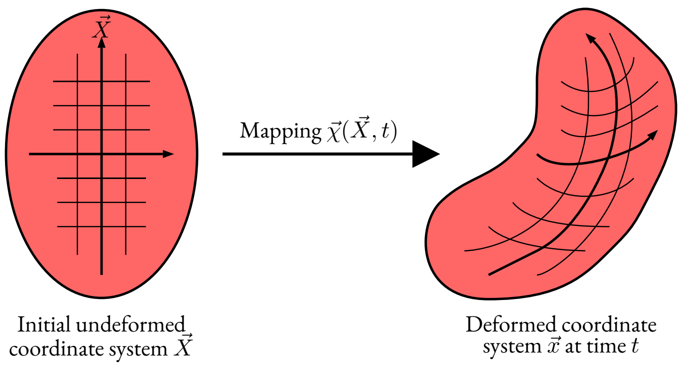
$\vec{X}$
Mapping $\vec{\chi}(\vec{X}, t)$
Initial undeformed coordinate system $\vec{X}$
Deformed coordinate system $\vec{x}$ at time t

Figure 1.1: Illustration of the hyperelastic solid deformation, where a time-dependent mapping $\vec{\chi}(\vec{X}, t)$ is applied to an initially undeformed solid with reference coordinate system $\vec{X}$ and results in a deformed coordinate system at time t.

The deformation gradient tensor $\mathbf{F}(\vec{X}, t)$ can also be expressed in terms of the reference map field $\vec{\xi}(\vec{x}, t)$ (Gurtin et al. 2010), which is an Eulerian mapping from the deformed coordinate system

$\vec{x}$ to the undeformed coordinate system $\vec{X}$. $\vec{\xi}(\vec{x},t)$ represents the inverse mapping of $\vec{\chi}$, *i.e.* $\vec{X} = \vec{\xi}(\vec{x},t) = \vec{\chi}^{-1}(\vec{x},t)$. The reference map field has an initialization of $\vec{\xi}(\vec{x},0) = \vec{x}$, and satisfies the advection equation with a material velocity $\vec{v}$,

$$\frac{\partial \vec{\xi}}{\partial t} + (\vec{v}\cdot\nabla)\vec{\xi} = \vec{0}. \tag{1.2}$$

By the chain rule, the deformation gradient tensor $\mathbf{F}(\vec{\xi},t)$ can be rewritten as

$$\mathbf{F}(\vec{\xi}(\vec{x},t),t) = \left(\frac{\partial \vec{\xi}(\vec{x},t)}{\partial \vec{x}}\right)^{-1}, \tag{1.3}$$

which can be used to describe the Cauchy stress of the solid $\boldsymbol{\sigma}$ in an Eulerian framework with a choice of constitutive relation $\mathbf{f}$. Here we choose $\mathbf{f}$ to be a compressible neo-Hookean elastic solid model (Rivlin & Saunders 1951), thus the solid stress can be denoted as

$$\boldsymbol{\sigma} = \mathbf{f}(\mathbf{F}) = GJ^{-5/3}\mathbf{B}' + \kappa(J-1)\vec{I}, \tag{1.4}$$

where the quantities in Eq. (1.4) are summarized in the Table 1.1.

Symbol	Meaning	Explanation
G	small-strain shear modulus	unit: Pa
κ	bulk modulus	$\approx 3G$
J	determinant of $\mathbf{F}$	$J = \det(\mathbf{F})$
$\mathbf{B}$	left Cauchy–Green tensor	$\mathbf{B} = \mathbf{F}\mathbf{F}^{\mathrm{T}}$
$\mathbf{B}'$	deviatoric part of $\mathbf{B}$	$\mathbf{B}' = \mathbf{B} - \frac{1}{3}\mathrm{trace}(\mathbf{B})\vec{I}$

Table 1.1: Symbols and their physical or mathematical meanings of the chosen neo-Hookean elastic solid model (Eq. (1.4)). An explicit evaluation equation or relevant information for each symbol is also listed.

This explicit form of the solid stress $\boldsymbol{\sigma}$ gives information to the the material velocity $\vec{v}(\vec{x},t)$ of the solid with density ρ_s, which satisfies the momentum equation

$$\rho_s \left(\frac{\partial \vec{v}}{\partial t} + (\vec{v} \cdot \nabla)\vec{v} \right) = \nabla \cdot \boldsymbol{\sigma}. \tag{1.5}$$

1.1.2 Overview of the Algorithm

The simulation domain is discretized by a fixed Eulerian mesh, whose nodes can be deformed to a new position according to the solid deformation. Assign a reference map field $\vec{\xi}$ to the mesh, we can track each nodal evolution and calculate the deformation gradient tensor $\mathbf{F}$ (Eq. (1.3)) at a given time t. With the choice of constitutive relation $\mathbf{f}$, the deformation gradient tensor $\mathbf{F}$ tells us the value of the solid stress $\boldsymbol{\sigma}$ (Eq. (1.4)). Combined with the momentum equation (Eq. (1.5)), the divergence of the solid stress $\nabla \cdot \boldsymbol{\sigma}$ formulates how the material velocity $\vec{v}$ changes with respect to time t. Starting at a timestep n, the RMT finds the material velocity for the next timestep $n+1$ based on the reference map. The algorithm of the velocity field and reference map field update is summarized below, followed by the discretization details in Chapter 2.

Algorithm 1: Update the solid kinematic fields using the RMT.

```
1 Begin
2     Initialize the velocity field $\vec{v}_0$ and the reference map field $\vec{\xi}_0$;
3     LOOP $\vec{v}^{\,n+1}, \vec{\xi}^{\,n+1} \leftarrow \vec{v}^{\,n}, \vec{\xi}^{\,n}$
4         Compute the reference map gradient (Eqs. (2.13) – (2.16));
5         Calculate the deformation gradient tensor $\mathbf{F}^n$;
6         Calculate the solid stress $\boldsymbol{\sigma}^n$;
7         Calculate the updated velocity field $\vec{v}^{\,n+1}$;
8         Calculate the updated reference map field $\vec{\xi}^{\,n+1}$ (Eq. (2.12)).
9 end
```

1.2 Lattice Boltzmann Method

1.2.1 Overview of the Lattice Boltzmann Method

Based on the kinetic theory of gases (Higuera & Jiménez 1989; Rivet & Boon 2005), the lattice Boltzmann (LB) method is an alternative to other traditional computational fluid dynamics methods, like the family of finite-difference, (Baliga & Patankar 1980; LeVeque 2007), finite-element (Hughes 2012), and finite-volume methods (LeVeque et al. 2002; Tryggvason et al. 2001), in the continuum realm. With a minimal form of the Boltzmann kinetic equation (Krüger et al. 2017), the LB method reconstructs the macroscopic hydrodynamic quantities, namely the density field ρ_f and the velocity field $\vec{v}$, of the Navier–Stokes equation with a mesoscopic representation of fluid particles. A mesoscopic probabilistic view of particle motion is introduced into the macroscopic hydrodynamics, and the latter is characterized by a probability distribution function $f(\vec{x}, \vec{v}, t)$, also known as the population, of a fluid particle with molecular velocity $\vec{v}$ being present at position $\vec{x}$ at a given time t. The macroscopic quantities can thus be retrieved as an integral in the velocity space (Succi 2001),

$$\rho_f(\vec{x}, t) = \int f(\vec{x}, \vec{v}, t)\, d\vec{v}, \tag{1.6}$$

which denotes the fluid density, and similarly, the local fluid velocity is denoted as

$$\vec{v}(\vec{x}, t) = \frac{\int f(\vec{x}, \vec{v}, t)\vec{v}\, d\vec{v}}{\rho_f(\vec{x}; t)}. \tag{1.7}$$

In a two-dimensional discrete space and time domain, the velocity space is reduced to nine discrete velocities (Shan et al. 2006), $\vec{c}_i$. This lattice model is referred to as D_2Q_9 (Qian et al. 1992). D_2 stands for two dimensions, and Q_9 represents the nine possible directions for a single particle to move, including the one staying at rest. For one lattice in such domain, there are nine populations f_i associated with it, each representing the probability distribution of the particle velocities in one timestep.

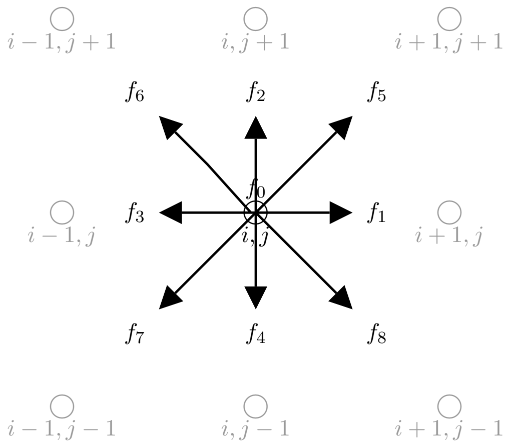

Figure 1.2: Diagram of the D_2Q_9 lattice model for the two-dimensional LB method. Each f_i is a probability distribution function of a particle velocity at (i, j) in the direction of the black arrow. The grey lattices are the neighboring lattices, which will receive the f_i's contribution in the next timestep.

The black arrows in Figure 1.2 represent the nine discrete velocities $\vec{c}_i$, which respectively indicate how a particle at (i, j) travels to its neighboring lattices. Since the LB method uses unit grid spacing and unit timestep in its discrete domain, *i.e.* $\Delta x = 1$ and $\Delta t = 1$, $\vec{c}_i = \dfrac{\Delta \vec{x}_i}{\Delta t}$ are denoted as

$$\begin{aligned} \vec{c}_1 &= (1,0) & \vec{c}_5 &= (1,1) \\ \vec{c}_2 &= (0,1) & \vec{c}_6 &= (-1,1) \\ \vec{c}_3 &= (-1,0) & \vec{c}_7 &= (-1,-1) \\ \vec{c}_4 &= (0,-1) & \vec{c}_8 &= (1,-1) \end{aligned} \tag{1.8}$$

with $\vec{c}_0 = (0,0)$ representing the stationary motion. The neighboring lattices associated with the velocities in the left column in Eq. (1.8) are called the first neighbors, and the ones with the right are the second neighbors. The one at rest, which is the particle at (i, j) itself is called the zeroth neighbor. There are different values of weight w_i associated with different level of neighbors, listed in Table 1.2.

neighbor f_i	0th	1st	2nd
number	1	4	4
weight w_i	$4/9$	$1/9$	$1/36$

Table 1.2: Attributes of the weights and number of each level of neighbors to one lattice in the D_2Q_9 model.

The sum of the nine weights of neighboring lattices is strictly one:

$$1 \times \frac{4}{9} + 4 \times \frac{1}{9} + 4 \times \frac{1}{36} = 1. \tag{1.9}$$

With the nine discrete velocities $\vec{c}_i$, the integral forms of macroscopic fluid quantities can be reduced to summations of the populations $f_i(\vec{x}, t)$. These summations correspond to the moments of probability distribution functions. The integral form of the macroscopic fluid density ρ_f in Eq. (1.6) becomes the zeroth moment of the populations f_i,

$$\rho_f(\vec{x}, t) = \sum_{i=0}^{8} f_i(\vec{x}, t). \tag{1.10}$$

The first moment corresponds with the integral form of the macroscopic fluid velocity in Eq. (1.7) and refers to the macroscopic fluid momentum $\vec{J}$,

$$\vec{J}(\vec{x}, t) = \rho_f(\vec{x}, t)\vec{v}(\vec{x}, t) = \sum_{i=0}^{8} \vec{c}_i f_i(\vec{x}, t), \tag{1.11}$$

and the macroscopic fluid velocity $\vec{v}$ is

$$\vec{v}(\vec{x}, t) = \frac{1}{\rho_f(\vec{x}, t)} \sum_{i=0}^{8} \vec{c}_i f_i(\vec{x}, t). \tag{1.12}$$

Suppose the fluid model can be described by a quasi-incompressible (Succi 2001; Krüger et al.

2017) Navier–Stokes equation using the macroscopic fluid quantities,

$$\partial_t \vec{v} + \vec{v} \cdot \nabla \vec{v} = \nu \Delta \vec{v} - \frac{1}{\rho_f} \nabla p \tag{1.13}$$

where $\vec{v}$ is the macroscopic fluid velocity, p is the fluid pressure, and ν is the kinematic viscosity under the incompressibility condition,

$$\nabla \cdot \vec{v} = 0. \tag{1.14}$$

Rather than directly discretizing this system of nonlinear partial differential equations in Eqs. (1.13) & (1.14), the LB method reconstructs the fluid density ρ_f and the local fluid velocity $\vec{v}$ by updating the populations $f_i(\vec{x}, t)$ based on the Bhatnagar–Gross–Krook (BGK) model (Bhatnagar et al. 1954),

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = f_i(\vec{x}, t) - \omega \left[f_i(\vec{x}, t) - f_i^{eq}(\vec{x}, t) \right] \tag{1.15}$$

where from left to right each term represents:

- f_i: the probability distribution function (population) of the fluid particle in the i-th direction,
- Δt: the timestep in the LB method, denoted to be $\Delta t = 1$,
- $\vec{c}_i$: the discrete velocity in the i-th direction, also known as the lattice velocity,
- ω: the relaxation frequency, $\omega = \dfrac{\Delta t}{\tau_{\text{LB}}}$,
    - τ_{LB} is the relaxation time to the local equilibrium, and can be modified in the simulation,
    - τ_{LB} controls the kinematic viscosity $\nu = \vec{c}_s^{\,2} \left(\tau_{\text{LB}} - \dfrac{\Delta t}{2} \right) = \vec{c}_s^{\,2} \left(\dfrac{\Delta t}{\omega} - \dfrac{\Delta t}{2} \right)$,
    - $\vec{c}_s$ is the speed of sound in the LB lattice unit, chosen to be $\vec{c}_s = \sqrt{\dfrac{1}{3}}$ (Eq. (A.4)),
- f_i^{eq}: the local Maxwell–Boltzmann equilibrium distribution function.

This local equilibrium distribution function f_i^{eq} is valid only when the particle is close to the Maxwell–Boltzmann equilibrium (Gombosi & Gombosi 1994), and can be approximated by a second-order Taylor expansion (Succi 2001),

$$f_i^{eq}(\vec{x},t) = f_i^{eq}(\rho,\vec{v}) = w_i\,\rho\left[1 + \frac{\vec{v}\cdot\vec{c}_i}{c_s^2} + \frac{(\vec{v}\cdot\vec{c}_i)^2 - c_s^2\vec{v}^2}{2c_s^4}\right]. \tag{1.16}$$

The second-order expansion is sufficient to get the momentum flux tensor, which contains the shear stress information. This f_i^{eq} matches the first two moments (Chapman & Cowling 1952), the fluid density ρ_f and velocity $\vec{v}$,

$$\begin{aligned} \rho_f(\vec{x},t) &= \sum_{i=0}^{8} f_i(\vec{x},t) = \sum_{i=0}^{8} f_i^{eq}(\vec{x},t), \\ \vec{J}(\vec{x},t) = \rho_f(\vec{x},t)\vec{v}(\vec{x},t) &= \sum_{i=0}^{8} \vec{c}_i f_i(\vec{x},t) = \sum_{i=0}^{8} \vec{c}_i f_i^{eq}(\vec{x},t), \end{aligned} \tag{1.17}$$

but generally does not match the second moment, which corresponds to the energy flux, unless at equilibrium. The second moment models how energy dissipates in viscous fluid, *i.e.* how the particle distribution moves away from the Maxwell–Boltzmann equilibrium. Therefore, the simulation parameters have to be within a certain range, or else the simulation will break down since the particle distribution is too far from the equilibrium, resulting in the probability distribution function at equilibrium f_i^{eq} being invalid. For example, the relaxation time τ_{LB} is typically chosen between $(0.5, 1.0]$. If below 0.5, the kinematic viscosity will be nonphysically negative, and if above 1.0, the fluid particle will be too far away from its equilibrium.

This BGK model in Eq. (1.15) can also be rearranged to generate two LB routines,

$$\underbrace{f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) - f_i(\vec{x},t)}_{\text{Streaming}} = -\,\underbrace{\omega\left[f_i(\vec{x},t) - f_i^{eq}(\vec{x},t)\right]}_{\text{collision}}. \tag{1.18}$$

The collision routine shows how the populations f_i of a particle interact amongst each other locally and how they move toward their Maxwell–Boltzmann equilibrium due to collisions. The momentum is conserved in the collision routine, whereas the kinetic energy is not, thus resulting in energy dissipation in the form of viscosity. The equation of the collision operator is

$$\Omega_i = -\omega(f_i - f_i^{eq}). \tag{1.19}$$

This collision operator can be further combined with f_i to calculate the post-collision population $\widehat{f_i}$,

$$\widehat{f_i} = f_i + \Omega_i \Delta t = (1-\omega) f_i + \omega f_i^{eq}, \tag{1.20}$$

which corresponds to the new population f_i at the corresponding neighbor for the next timestep.

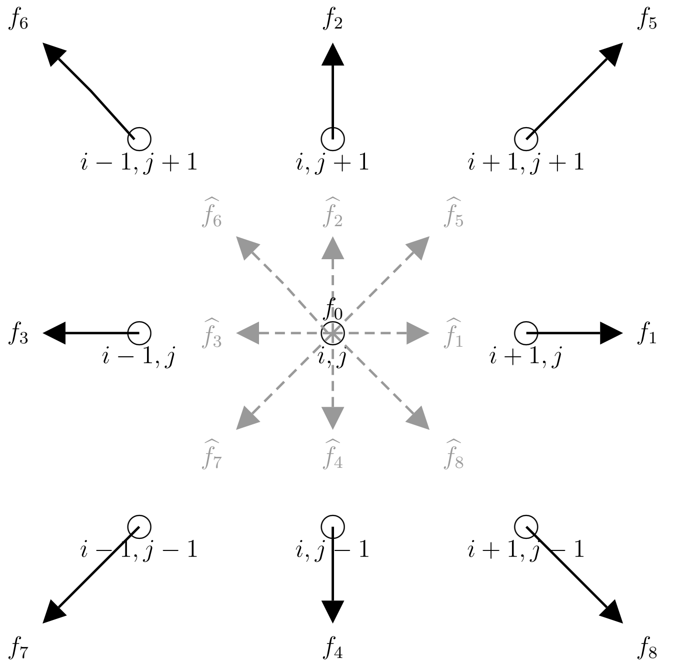

Figure 1.3: Diagram of the streaming routine for the two-dimensional LB method. The grey arrows and labels show the original position of the nine post-collision populations $\widehat{f_i}$. Each $\widehat{f_i}$ moves from its grey position in the direction of dashed arrow to its neighboring lattice. $\widehat{f_i}$ then becomes the new f_i of the neighboring lattices in the next timestep.

In the streaming routine, illustrated in Figure 1.3, $\widehat{f}_p$ moves forward to its neighboring lattice along the $\vec{c}_i$ direction, and its value is taken to update the value of f_i for the next timestep. There is no information lost in the process since streaming is local and exact up to machine precision. The equation for the streaming routine is

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = \widehat{f}_i(\vec{x}, t). \tag{1.21}$$

Eq. (1.18) and Eq. (1.21) are first-order updates in time since the updated term equals one timestep multiplying the direct change. Eq. (1.21) is also the Navier–Stokes equation in the spirit of the LB method (Succi 2001). Since the LB method steps forward with a unity timestep, $\Delta t = 1$, the LB method can be viewed as a finite-difference method looped over a fixed Eulerian simulation domain where nine populations are associated with each node. This feature makes the LB method easy to implement and parallelize (Kandhai et al. 1998; Amati et al. 1997; Bernaschi et al. 2009) since the code base is characterized by two nested loops.

1.2.2 Overview of the Algorithm

On a two-dimensional simulation domain discretized by a fixed Eulerian mesh, we can track the evolution of the hydrodynamic terms, in particular the fluid density ρ_f and velocity $\vec{v}$, with the moments of the populations f_i attributed to each node. In one timestep, we first calculate the f_i at equilibrium f_i^{eq} with known fluid density ρ_f and velocity $\vec{v}$ from the previous timestep. Coupled with f_i, we use this f_i^{eq} in the collision routine to compute the post-collision distribution $\widehat{f}_i$. In the streaming routine, the $\widehat{f}_i$ moves to its neighboring lattice in the direction of $\vec{c}_i$, and becomes the new f_i of the neighboring lattice in the next timestep. Then in the next timestep, we recalculate the zeroth and first moments to retrieve the updated fluid density ρ_f and velocity $\vec{v}$. The algorithm of the LB method is summarized below, with the four key routines in the loop (`equilibrium`, `collision`, `streaming`, and `hydro`). More details regarding how to incorporate external forces (Guo et al. 2002) and adjust the discretization to be second-order in time (Krüger et al. 2017), how to apply boundary conditions (Succi 2001) and how to initialize the f_i (Krüger et al. 2017) will be discussed in Chapter 2.

Algorithm 2: Update the hydrodynamic fields using the LB method.

```
1 Begin
2     Initialize the fluid density field $\rho_f$ , velocity field $\vec{v}_0$, and $f_i$;
3     LOOP $f_i^{n+1}, \rho_f^{n+1}, \vec{v}^{\,n+1} \leftarrow f_i^{\,n}, \rho_f^n, \vec{v}^{\,n}$
4         equilibrium: Calculate $f_i^{eq\ n}$ using $\rho_f^n$ and $\vec{v}^{\,n}$ (Eq. (1.16));
5         collision: Calculate $\widehat{f_i}^{\,n}$ as a result of local $f_i^n$ and $f_i^{eq\ n}$ collision (Eq. (1.20));
6         streaming: Update $f_i^{n+1}$ of the neighboring lattices with $\widehat{f_i}^{\,n}$ (Eq. (1.21));
7         hydro: Calculate $\rho_f^{n+1}$ (Eq. (1.10)) and $\vec{v}^{\,n+1}$ (Eq. (1.12));
8 end
```

1.3 Fluid–Structure Interaction

The configuration of the fluid–structure interaction modelled in the thesis is a deformable solid immersed within the fluid. This compressible neo-Hookean (Eq. (1.4)) solid can bend, twist and move with the fluid flow. A level set function $\phi(\vec{x}, t)$ (Sethian 1999; Osher & Sethian 1988) is added to the configuration to describe the solid boundary. It is also used to label the phase of each node (fluid, solid, or within the extrapolation zone) based on the signed distance function values,

$$\text{label of node} = \begin{cases} \text{solid}, & \text{if } \phi < 0 \\ \text{fluid}, & \text{if } \phi > \epsilon \\ \text{within extrapolation zone}, & \text{if } 0 \leq \phi \leq \epsilon \end{cases} \tag{1.22}$$

The reference map $\vec{\xi}$ is only defined in the solid region. The extrapolation zone of width ϵ links the purely solid and purely fluid region, and the nodes within the extrapolation zone can switch between the two phases depending on their reference map values and the level set function. In order to obtain the reference map within the extrapolation zone, a least-squares regression procedure is used to obtain the extrapolated reference map values (Rycroft et al. 2018)—see Chapter 2 for further details.

This fluid–structure interaction configuration involves the following quantities in their respective regions of solid only, fluid only, and global in both phases:

Global	
velocity field $\vec{v}$	
Solid	Fluid
solid density ρ_s	fluid density ρ_f
solid stress $\boldsymbol{\sigma}$	fluid kinematic viscosity ν
reference map field $\vec{\xi}$	

Table 1.3: Quantities and their valid regions involved in the FSI simulation. The velocity field $\vec{v}$ is define in both the fluid and the solid region. The solid density ρ_s, solid stress σ and reference map field $\vec{\xi}$ are only denoted in the solid region; whereas the fluid density ρ_f and fluid kinematic viscosity ν only in the fluid.

The entire simulation domain is discretized with a fixed mesh, where the solid nodes are attributed with the global velocity $\vec{v}$, the solid density ρ_s, the reference map field $\vec{\xi}$ and the solid stress $\boldsymbol{\sigma}$, and the fluid nodes are attributed with the global velocity $\vec{v}$, the fluid density ρ_f and the fluid kinematic viscosity ν. The extrapolation zone is updated at each timestep and extrapolated to refresh the solid–fluid interface. Every node is assigned with the nine populations f_i to process the LB routines. Since each node carries mesoscopic information of the populations for both the fluid and the solid, the new hybrid method will not be requiring separate routines to calculate the solid density ρ_s and the fluid density ρ_f. Their calculations can be unified using the zeroth moment of the particle populations (Eq. (1.10)). Due to this setup, the hybrid method does not need to abide by the region differentiation

(Table 1.3) when calculating the FSI quantities. It extends the mesoscopic view from the fluid to the solid, resulting in a unified representation of the entire simulation domain. The thesis proposes to use the RMT for the solid stress $\boldsymbol{\sigma}$ and solid reference map $\vec{\xi}$ update, and the LB method for the fluid density ρ_f, solid density ρ_s, and global velocity $\vec{v}$ update for FSI simulations. Compared with the previous RMT-based FSI numerical methods (Kamrin et al. 2012; Valkov et al. 2015; Rycroft et al. 2018), an explicit calculation of the fluid stress for the FSI simulation also will not be needed in this new hybrid method. The fluid stress has been built into the LB equilibrium equation (Succi 2001) in the BGK model (Eq. (1.18)), which is automatically calculated if performing the entire LB algorithm.

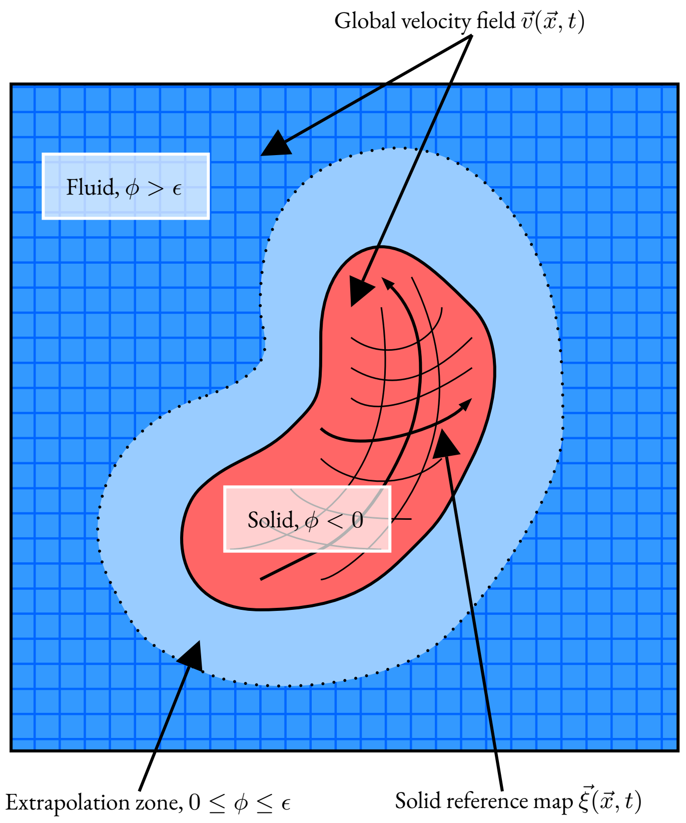
Global velocity field $\vec{v}(\vec{x},t)$
Fluid, $\phi > \epsilon$
Solid, $\phi < 0$
Extrapolation zone, $0 \le \phi \le \epsilon$
Solid reference map $\vec{\xi}(\vec{x},t)$

Figure 1.4: Illustration of the reference map configuration for a fluid–structure interaction with a deformable solid immersed in fluid on a fixed Eulerian grid. A signed distance function denotes the level set function $\phi(\vec{x},t)$, where a positive distance represents the fluid and a negative represents the solid. The extrapolation zone generates a smooth transition between the two phases, defined as $0 \le \phi \le \epsilon$ where ϵ is the width of the extrapolation zone.

2

Lattice-Boltzmann-Based Reference Map Technique

The key to combine the lattice Boltzmann method (LBM) and the reference map technique (RMT) into one fully-Eulerian method is to forward the RMT-based solid stress as the external force density into the LB-based routines. Solid deformation can be quantified by the solid stress, whose expression is consistent with the force density. Since the LB method supports external forcing (Guo et al. 2002; Huang et al. 2011; Ginzburg et al. 2008; Shan & Chen 1993; He et al. 1998), it can theoretically accept a solid stress and extend the notion of mesoscopic fluid motion to mesoscopic solid deformation. This is realized by modifying the collision operator and the moments of the particle populations in the LB method. The resulting lattice-Boltzmann-based reference map technique (LB-RMT) for a fluid–structure interaction simulation utilizes its parent methods respectively:

- the RMT tracks the solid reference map $\vec{\xi}$ and the solid stress $\boldsymbol{\sigma}$,
- the LB method tracks the global velocity $\vec{v}$, the fluid density ρ_f, and the solid density ρ_s.

2.1 Lattice-Boltzmann-Based Solid Deformation

Imagine that a lattice in the LB domain now represents a solid, if we are to directly apply the usual LB method, the simulation will lack the information regarding the solid deformation. In order to have an accurate kinematical description of the solid lattice, we need to pass in the information of the solid forces into the LB routines. Let us first simplify our simulation domain to be solid only to derive the theory of the LB-based solid deformation. In the RMT, the material velocity $\vec{v}$ satisfies the macroscopic momentum equation

$$\rho_s \frac{D\vec{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma}, \tag{2.1}$$

where $\boldsymbol{\sigma}$ is the solid stress and ρ_s is the solid density. The momentum (Eq. (1.11)) or the material velocity $\vec{v}$ in the LB method equivalently satisfies the macroscopic momentum equation

$$\rho_f \frac{D\vec{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma}_f \tag{2.2}$$

where $\boldsymbol{\sigma}_f$ represents the fluid stress and ρ_f is the fluid density. For a simulation node in the LB method, it represents the fluid whose macroscopic hydrodynamics can be mesoscopically retrieved using the moments of particle populations, including the fluid stress $\boldsymbol{\sigma}_f$ (Succi 2001). In order to extend the attribute of the simulation node to contain solid, *i.e.* rewrite Eq. (2.1) in the language of the LB method, the only change is to add the solid stress into the LB routines. Since all the macroscopic hydrodynamic quantities are expressed in terms of f_i in the LB method, we can simply add the correction of the solid stress $\nabla \cdot \boldsymbol{\sigma}$ to the current LB first moment. One direct modification is to change the first moment and the collision operator. Adding the correction of the divergence of the solid stress to Eq. (1.11) in the LB method results in

$$\vec{J} = \nabla \cdot \boldsymbol{\sigma} + \sum_{i=0}^{8} \vec{c}_i f_i(\vec{x}, t). \tag{2.3}$$

In order to reflect this net change in the momentum equation on each individual population f_i, we can modify the collision operator from Eq. (1.19) so that each f_i will now carry the information of the solid stress,

$$\Omega_i = -\omega \left(f_i - f_i^{eq}\right) + w_i \frac{\nabla \cdot \boldsymbol{\sigma}}{c_s^2} \cdot \vec{c}_i. \tag{2.4}$$

This preliminary connection between the LB method and RMT is summarized in the table below:

	Material Velocity	First Moment	Collision Operator
RMT	$\rho \frac{D\vec{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma}$		
LBM	$\rho \frac{D\vec{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma}_f$	$\sum_{i=0}^{8} \vec{c}_i f_i(\vec{x}, t)$	$\Omega_i = -\omega \left(f_i - f_i^{eq}\right)$
LB-RMT	$\rho \frac{D\vec{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma} + \nabla \cdot \boldsymbol{\sigma}_f$	$\nabla \cdot \boldsymbol{\sigma} + \sum_{i=0}^{8} \vec{c}_i f_i(\vec{x}, t)$	$\Omega_i = -\omega \left(f_i - f_i^{eq}\right) + w_i \frac{\nabla \cdot \boldsymbol{\sigma}}{c_s^2} \cdot \vec{c}_i$

Table 2.1: Overview of the comparison between the RMT and LB method, and the derivation of the LB-RMT using the solid stress correction in the first moment, and the resulting collision operator.

Note that in this modification, each solid node inherits a fluid stress that is built into the LB method in the equilibrium step (Succi 2001), in addition to its solid stress. This automatically leads to additional viscous stress in the solid, which is in fact desired in the RMT implementation. The previous RMT

(Rycroft et al. 2018) incorporates an artificial viscous stress inside the solid to stabilize the simulation. By mesoscopically calculating the solid stress, the LB-based solid deformation stabilizes the simulation with a built-in viscous stress in the solid at no additional computation cost.

2.2 Modification of the Lattice Boltzmann Method with Forces

Mathematically speaking, directly adding the correction term to the first moment and the collision operator (Table 2.1) in the LB method does not violate the continuity equation and the momentum equation. However, this crude modification will result in unstable simulations with uncontrollable noise caused by discrete lattice artifacts (Krüger et al. 2017). Since the solid stress is analogous to the force density, a promising alternative is to add forces to the LB method (Guo et al. 2002).

A general approach to incorporating any arbitrary force density into the LB method requires a second-order accuracy in the velocity space and time to avoid instability. The standard LB method summarized in Chapter 1 is only first-order in velocity space (Eq. (1.12)) and in time (Eq. (1.19)). With specific modifications to two LB routines, `collision` and `hydro`, forces become present in the LB method. The macroscopic density and velocity, *i.e.* the zeroth and first moment of the particle populations, of a fluid particle now contain the half-force correction (Krüger et al. 2017),

$$\begin{aligned}\rho &= \sum_i f_i + \frac{\Delta t}{2}\sum_i F_i \\ \vec{v} &= \frac{1}{\rho}\sum_i f_i\vec{c}_i + \frac{\Delta t}{2\rho}\sum_i F_i\vec{c}_i.\end{aligned} \tag{2.5}$$

This new definition of velocity $\vec{v}$, which is of second-order in velocity space, is the physical velocity of the fluid particle, taken to be the average velocity of the pre-collision and post-collision values. The collision operator also becomes second-order in time, with a change of variable $\bar{\tau} = \tau + \frac{\Delta t}{2}$ (Dellar

2001; Chen & Doolen 1998),

$$\Omega_i = -\frac{1}{\bar{\tau}}(f_i - f_i^{eq}) + \left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_i, \tag{2.6}$$

and the post-collision population is also changed to a second-order expression with $\bar{\omega} = \frac{1}{\bar{\tau}}$,

$$\widehat{f_i} = f_i + \Omega_i \Delta t = (1 - \bar{\omega}) f_i + \bar{\omega} f_i^{eq} + \left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_i. \tag{2.7}$$

Suppose the external force density is denoted as $\vec{F_i}$ macroscopically, its mesoscopic definition F_i in the nine discrete velocity space takes the form

$$F_i = w_i \left(\frac{\vec{c_i} - \vec{v}}{c_s^2} + \frac{(\vec{c_i} \cdot \vec{v})\vec{c_i}}{c_s^4} \right) \cdot \vec{F}, \tag{2.8}$$

and in the context of the fluid–structure interaction, the force density is the solid stress, *i.e.* $\vec{F} = \nabla \cdot \sigma$.

	First-Order Standard LB Method	Second-Order LB Method with Forces
`hydro`	$\rho = \sum_i f_i$ $\vec{v} = \frac{1}{\rho} \sum_i f_i \vec{c_i}$	$\rho = \sum_i f_i + \frac{\Delta t}{2} \sum_i F_i$ $\vec{v} = \frac{1}{\rho} \sum_i f_i \vec{c_i} + \frac{\Delta t}{2\rho} \sum_i F_i \vec{c_i}$
`equilibrium`	$f_i^{eq} = w_i\, \rho \left[1 + \frac{\vec{v} \cdot \vec{c_i}}{c_s^2} + \frac{(\vec{v} \cdot \vec{c_i})^2 - c_s^2 \vec{v}^2}{2 c_s^4} \right]$	
`collision`	$\Omega_i = -\frac{1}{\tau}(f_i - f_i^{eq})$	$\Omega_i = -\frac{1}{\bar{\tau}}(f_i - f_i^{eq}) + \left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_i$
`streaming`	$f_i(\vec{x} + \vec{c_i}\Delta t, t + \Delta t) = \widehat{f_i}(\vec{x}, t) = f_i + \Omega_i \Delta t$	

Table 2.2: Comparisons in the main routines between the first-order LB method and the second-order LB method with external forces. The `equilibrium` and `stream` routines are left unchanged in the force modification, with exceptions in the half-time correction in the `hydro` and external force density in `collision` routines.

2.3 Overview of the Algorithm

With the new collision routine of the LB-based solid deformation, the hybrid method connects the two Eulerian-based methods together onto one fixed background mesh. For the FSI simulation configuration shown in Figure 1.4, the simulation starts with an initial global velocity field $\vec{v}_0$, initial densities of both fluid ρ_f and solid ρ_s, an initial solid reference map field $\vec{\xi}_0$, and a level set function ϕ that describes the solid–fluid interface. The RMT calculates the solid stress $\boldsymbol{\sigma}$ based on the reference map deformation gradient tensor, and this solid stress is then passed to the solid nodes as force density. The LB method updates the density fields, ρ_f and ρ_s, and the global velocity field $\vec{v}$ with the corresponding routines on the fluid nodes and the solid nodes. After the kinematic fields update, the reference map is extrapolated to the extrapolation zone to update the solid–fluid interface and relabel the solid and fluid nodes based on the new signed distance. The key is to shuffle the ordering of routines in Algorithm 1 and Algorithm 2. Here we represent the algorithm of the lattice-Boltzmann-based reference map technique (LB-RMT) in Algorithm 3, where blue represents the LB-native routines, red represents the RMT-native routines, and purple represents hybrid routines.

2.4 Numerical Method

The simulation domain (Figure 2.1) has length L and height H. Denote n_x grid points along the x axis and n_y grid points along the y axis, with grid spacing of unit length $\Delta x = \Delta y = 1$. There are $n_x \times n_y$ physically-defined nodes, among which fluid nodes are blue and solid nodes are red. Two extra layers of buffers are padded to each boundary in order to perform the second-order upwinding method and impose boundary conditions, leading the total number of nodes in the grid to be $(n_x+4)\times(n_y+4)$. The solid–fluid interface is represented by a purple line, which is surrounded by an extrapolation zone. The extrapolation zone has width $\epsilon = 5$, *i.e.* five layers. The nodes in each layer are represented with a different color and a corresponding indicator value of their level number.

Algorithm 3: LB-RMT of FSI with extradpolation zone.

```
1  Begin
2      Initialize the global velocity field $\vec{v}_0$, fluid density $\rho_f$ , solid density $\rho_s$ , and $f_i$ ;
3      Label the solid and fluid nodes using the level set function $\phi(\vec{\xi}_0)$ ;
4      Initialize the solid reference map $\vec{\xi}_0$ ;
5      LOOP $\rho_f^{n+1}, \rho_s^{n+1}, \vec{v}^{\,n+1}, \vec{\xi}^{\,n+1}, f_i^{\,n+1} \leftarrow \rho_f^n, \rho_s^n, \vec{v}^{\,n}, \vec{\xi}^{\,n}, \widehat{f_i}^{\,n}, \phi$
6          Update the reference map $\vec{\xi}^{\,n+1}$ (Eq. (2.10));
7          Extrapolate the reference map values in the extrapolation zone ;
8          Relabel the fluid and solid nodes in the extrapolation zone using $\phi(\vec{\xi}^{\,n+1})$;
9          Compute the reference map gradient (Eqs. (2.13)–(2.16)) ;
10         Calculate the deformation gradient tensor $\mathbf{F}^{n+1}$;
11         Calculate the divergence of solid stress $\nabla \cdot \boldsymbol{\sigma}^{n+1}$ (Eq. (2.17));
12         equilibrium: Calculate $f_i^{eq\ n}$ using $\rho_f^n$ or $\rho_s^n$ and $\vec{v}^{\,n}$ (Eq. (2.19)) ;
13         collision: Calculate $\widehat{f_i}^{\,n}$ as a result of local $f_i^{\,n}$ and $f_i^{eq\ n}$ collision (Eq. (2.6)) ;
14         bc: Apply the boundary conditions;
15         streaming: Update $f_i^{\,n+1}$ of the neighboring lattices with $\widehat{f_i}^{\,n}$ (Eq. (2.20));
16         hydro: Update $\rho_f^{n+1}$, $\rho_s^{n+1}$ , and $\vec{v}^{\,n+1}$ (Eq. (2.23));
17 end
```

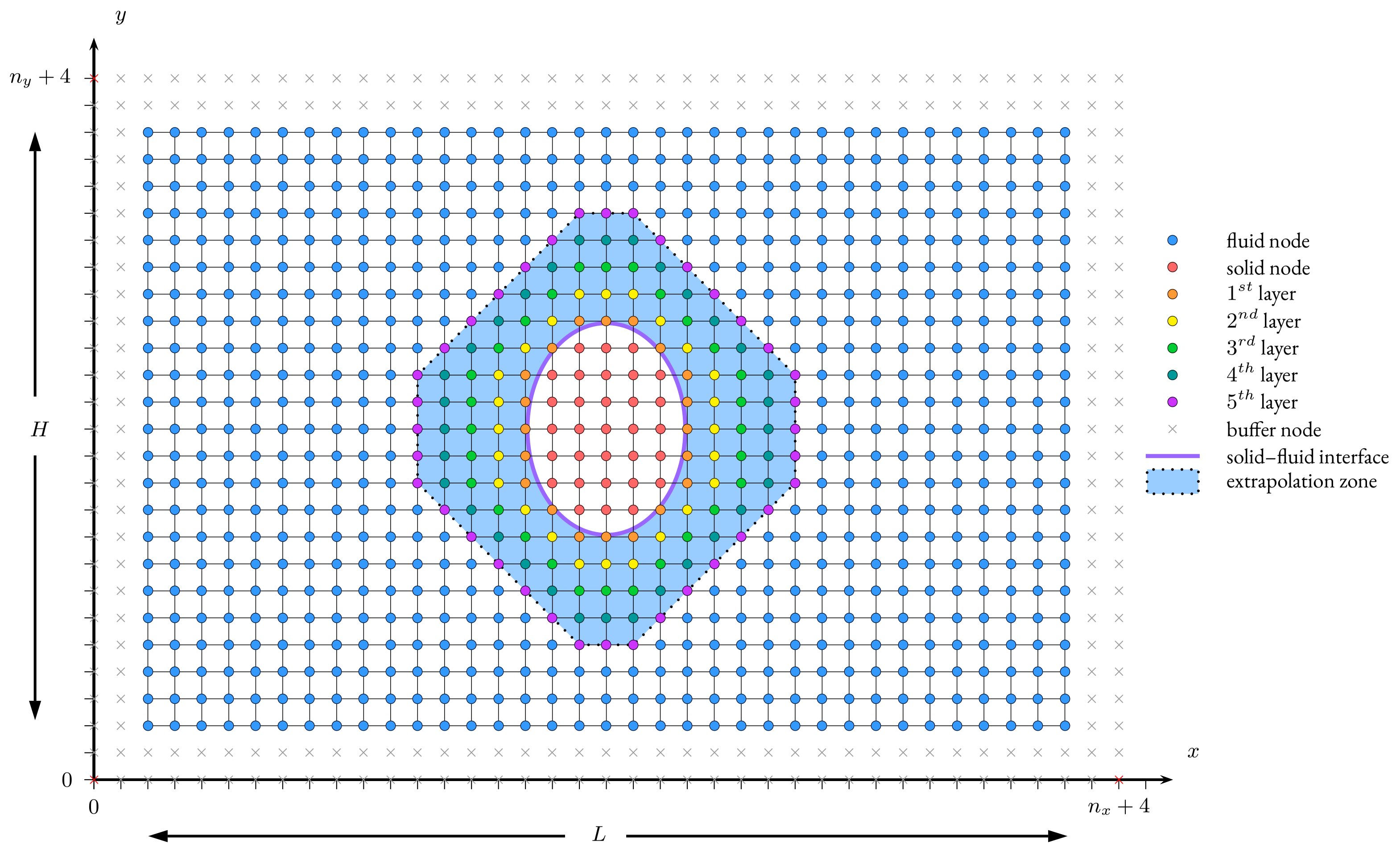

Figure 2.1: Grid setup of the lattice-Boltzmann-based reference map technique for a two-dimensional fluid–structure interaction simulation. The simulation domain has length L and height H, with $n_x \times n_y$ physical nodes. Two layers of buffer nodes are padded to each side, giving a total of $(n_x + 4) \times (n_y + 4)$ nodes in the grid. Solid nodes are represented in red, and fluid nodes are in blue. The solid–fluid interface is marked by the purple line, and the extrapolation zone between the two phases is colored in light blue. There are five layers in the extrapolation zone based on their distances to the interface. The nodes in each layer are assigned with a corresponding indicator value, *e.g.* nodes in the first layer are given an indicator of value 1.

2.4.1 Initial Condition

A common choice to initialize the macroscopic fluid quantities in the LB method is to set the initial density to be uniform and the initial velocity to be zero or some constant value (Krüger et al. 2017). Once ρ_0 and $\vec{v}_0$ are initialized, a prevalent modeling choice is to initialize the populations of each lattice $f_i (i = 0, \ldots, 8)$ to the equilibrium values implied by these macroscopic variables,

$$f_i(\vec{x}, 0) = f_i^{eq}(\rho_0, \vec{v}_0). \tag{2.9}$$

Here the method is modified to initialize the density and velocity fields of the fluid and the solid respectively. Therefore, for fluid lattices, the initial density ρ_0 is set to ρ_f , and solid lattices, to ρ_s .

2.4.2 Reference Map Update

Eq. (1.2) gives an update rule of the reference map $\vec{\xi}$. Suppose $\vec{\xi} = (X, Y)$ and $\vec{v} = (u, v)$, this advection equation of the reference map can be rewritten as

$$\vec{\xi}_{i,j}^{n+1} = \vec{\xi}_{i,j}^{n} - (u \cdot \partial_x + v \cdot \partial_y)\vec{\xi}_{i,j}^{n}. \tag{2.10}$$

A second-order ENO upwinding method (Udaykumar et al. 2003; Shu 1998) is used for discretization on the reference map for better accuracy. The second-order upwinding scheme is

$$\begin{aligned} \frac{\partial \vec{\xi}_{i,j}}{\partial x} &= \begin{cases} \dfrac{3\vec{\xi}_{i,j} - 4\vec{\xi}_{i-1,j} + \vec{\xi}_{i-2,j}}{2\Delta x} & \text{if } u > 0 \\ \dfrac{-3\vec{\xi}_{i,j} + 4\vec{\xi}_{i+1,j} - \vec{\xi}_{i+2,j}}{2\Delta x} & \text{if } u < 0 \end{cases}, \\ \frac{\partial \vec{\xi}_{i,j}}{\partial y} &= \begin{cases} \dfrac{3\vec{\xi}_{i,j} - 4\vec{\xi}_{i,j-1} + \vec{\xi}_{i,j-2}}{2\Delta y} & \text{if } v > 0 \\ \dfrac{-3\vec{\xi}_{i,j} + 4\vec{\xi}_{i,j+1} - \vec{\xi}_{i,j+2}}{2\Delta y} & \text{if } v < 0 \end{cases}. \end{aligned} \tag{2.11}$$

Each component of $\vec{\xi}_{i,j}^{n+1} = (X_{i,j}^{n+1}, Y_{i,j}^{n+1})$ can be calculated respectively based the update rule (Eq. (2.10)) and the discretized gradient of the reference map (Eq. (2.11)),

$$\begin{aligned} X_{i,j}^{n+1} &= X_{i,j}^{n} - \left(u\frac{\partial X}{\partial x} + v\frac{\partial X}{\partial y}\right) \\ Y_{i,j}^{n+1} &= Y_{i,j}^{n} - \left(u\frac{\partial Y}{\partial x} + v\frac{\partial Y}{\partial y}\right) \end{aligned} \tag{2.12}$$

2.4.3 Extrapolation in the Extrapolation Zone

The extrapolation routine is based on fitting a least-square regression (Rycroft et al. 2018) instead of a conventional PDE-based method (Aslam 2004; Rycroft & Gibou 2012; Valkov et al. 2015). This alternative reduces the complexity in keeping track of the level set ϕ and the reference map $\vec{\xi}$ values.

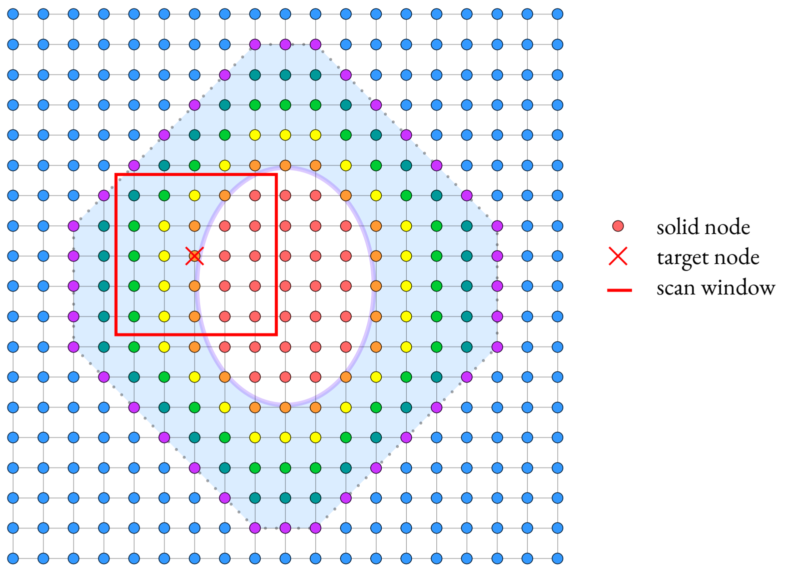

Figure 2.2: Illustration of the reference map extrapolation in the extrapolation zone. The extrapolation starts from the first layer in the extrapolation zone then moves outward to the next layer after all nodes have been extrapolated. A scan window of width $r = 5$ centered at the target node is initialized, and the extrapolated reference map is calculated using the solid nodes inside the scan window. If the result is invalid, then increase the scan window width r to include more solid nodes.

Start at node (i, j) in the first layer of the extrapolation zone, the extrapolation procedure is:

1. Initialize a scan window with an initial width of $r = 5$ centered at (i, j) and count the number of solid nodes. If there are less than two solid nodes, increase the width r, and repeat Step 1.
2. Use least-square regression to fit a linear map $\vec{\xi}_{\text{extrap}}(x, y) = Ax + By + C$ using the available reference map values of the enclosed solid nodes and their positions. If the linear map is ill-defined, increase the width r and repeat Step 1 and 2.
3. Assign $\vec{\xi}_{\text{extrap}}$ to be the reference map values of node (i, j).

Once all nodes in the first layer have been processed, move outward to the next level in the extrapolation zone. After the extrapolation procedure, relabel the nodes in the extrapolation zone by calculating their level set values. The extrapolation zone offers a smooth transition between the solid and the fluid, requiring no additional bookkeeping about density, velocity, or no-slip solid–fluid interface.

2.4.4 Divergence of the Solid Stress

The purpose of Step 8, 9, 10 in Algorithm 3 is to calculate the divergence of the solid stress in Step 11, which to be used in the `collision` routine for solid nodes. The divergence of solid stress $\nabla \cdot \boldsymbol{\sigma}_{i,j}$ is calculated based on the four intermediate solid stresses $\boldsymbol{\sigma}_{i-\frac{1}{2},j}$, $\boldsymbol{\sigma}_{i+\frac{1}{2},j}$, $\boldsymbol{\sigma}_{i,j-\frac{1}{2}}$, $\boldsymbol{\sigma}_{i,j+\frac{1}{2}}$ to the left, right, bottom, and top respectively. Each intermediate solid stress is computed from the Jacobian of the reference map $\vec{\xi}$ with a second-order finite difference scheme (Rycroft et al. 2018). The gradient involved to calculate the Jacobian for the left solid stress $\boldsymbol{\sigma}_{i-\frac{1}{2},j}$ is

$$\left(\frac{\partial \vec{\xi}}{\partial x}\right)_{i-\frac{1}{2},j} = \frac{\vec{\xi}_{i,j} - \vec{\xi}_{i-1,j}}{\Delta x}, \left(\frac{\partial \vec{\xi}}{\partial y}\right)_{i-\frac{1}{2},j} = \frac{\vec{\xi}_{i,j+1} + \vec{\xi}_{i-1,j+1} - \vec{\xi}_{i,j-1} - \vec{\xi}_{i-1,j-1}}{4\Delta y}. \quad (2.13)$$

The gradient involved to calculate the Jacobian for the right solid stress $\boldsymbol{\sigma}_{i+\frac{1}{2},j}$ is

$$\left(\frac{\partial \vec{\xi}}{\partial x}\right)_{i+\frac{1}{2},j} = \frac{\vec{\xi}_{i+1,j} - \vec{\xi}_{i,j}}{\Delta x}, \left(\frac{\partial \vec{\xi}}{\partial y}\right)_{i+\frac{1}{2},j} = \frac{\vec{\xi}_{i+1,j+1} + \vec{\xi}_{i,j+1} - \vec{\xi}_{i+1,j-1} - \vec{\xi}_{i,j-1}}{4\Delta y}. \quad (2.14)$$

The gradient involved to calculate the Jacobian for the bottom solid stress $\boldsymbol{\sigma}_{i,j-\frac{1}{2}}$ is

$$\left(\frac{\partial\vec{\xi}}{\partial x}\right)_{i,j-\frac{1}{2}} = \frac{\vec{\xi}_{i+1,j} + \vec{\xi}_{i+1,j-1} - \vec{\xi}_{i-1,j} - \vec{\xi}_{i-1,j-1}}{4\Delta x}, \left(\frac{\partial\vec{\xi}}{\partial y}\right)_{i,j-\frac{1}{2}} = \frac{\vec{\xi}_{i,j} - \vec{\xi}_{i,j-1}}{\Delta y}. \quad (2.15)$$

The gradient involved to calculate the Jacobian for the top solid stress $\boldsymbol{\sigma}_{i,j+\frac{1}{2}}$ is

$$\left(\frac{\partial\vec{\xi}}{\partial x}\right)_{i,j+\frac{1}{2}} = \frac{\vec{\xi}_{i+1,j+1} + \vec{\xi}_{i+1,j} - \vec{\xi}_{i-1,j+1} - \vec{\xi}_{i-1,j}}{4\Delta x}, \left(\frac{\partial\vec{\xi}}{\partial y}\right)_{i,j+\frac{1}{2}} = \frac{\vec{\xi}_{i,j+1} - \vec{\xi}_{i,j}}{\Delta y}. \quad (2.16)$$

Each Jacobian is then coupled with the constitutive relation $\mathbf{f}$ to derive the intermediate solid stress. Once the four intermediate stresses are computed, the divergence of solid stress at (i, j) is denoted as

$$\nabla \cdot \boldsymbol{\sigma} = \frac{(\boldsymbol{\sigma}_{i+\frac{1}{2},j})_x - (\boldsymbol{\sigma}_{i-\frac{1}{2},j})_x}{\Delta x} + \frac{(\boldsymbol{\sigma}_{i,j+\frac{1}{2}})_y - (\boldsymbol{\sigma}_{i,j-\frac{1}{2}})_y}{\Delta y}. \quad (2.17)$$

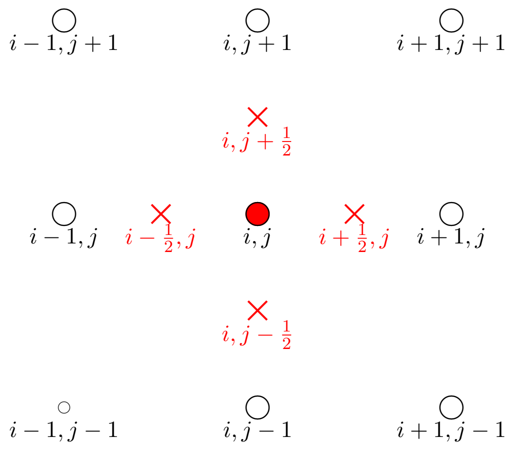

Figure 2.3: Stencil setup for the calculation the divergence of solid stress. The red circle represents the current solid node, and the red crosses represent the mid-grid intermediate solid stresses. To compute these stresses, for example the left solid stress $(\sigma_s)_{i-\frac{1}{2},j}$, six solid nodes in total are involved in the gradient calculations: $(i, j), (i-1, j)$ for the x-direction derivatives, and $(i, j+1), (i-1, j+1), (i, j-1), (i-1, j-1)$ for the y-direction derivatives.

2.4.5 Equilibrium

The equilibrium step is local to each fluid lattice,

$$f_i^{eq} = w_i \, \rho \left[1 + \frac{\vec{v} \cdot \vec{c}_i}{c_s^2} + \frac{(\vec{v} \cdot \vec{c}_i)^2 - c_s^2 \vec{v}^2}{2 c_s^4} \right], \tag{2.18}$$

and explicitly, each f_i is expanded as

$$\begin{aligned}
f_0^{eq}(i,j) &= \rho w_0 \left(1 - \frac{u^2+v^2}{2\vec{c}_s^2} \right) \\
f_1^{eq}(i,j) &= \rho w_1 \left(1 + \frac{u}{\vec{c}_s^2} + \frac{u^2}{2\vec{c}_s^4} - \frac{u^2+v^2}{2\vec{c}_s^2} \right) \\
f_2^{eq}(i,j) &= \rho w_2 \left(1 + \frac{v}{\vec{c}_s^2} + \frac{v^2}{2\vec{c}_s^4} - \frac{u^2+v^2}{2\vec{c}_s^2} \right) \\
f_3^{eq}(i,j) &= \rho w_3 \left(1 - \frac{u}{\vec{c}_s^2} + \frac{u^2}{2\vec{c}_s^4} - \frac{u^2+v^2}{2\vec{c}_s^2} \right) \\
f_4^{eq}(i,j) &= \rho w_4 \left(1 - \frac{v}{\vec{c}_s^2} + \frac{v^2}{2\vec{c}_s^4} - \frac{u^2+v^2}{2\vec{c}_s^2} \right) \\
f_5^{eq}(i,j) &= \rho w_5 \left(1 + \frac{u}{\vec{c}_s^2} + \frac{v}{\vec{c}_s^2} + \frac{uv}{\vec{c}_s^4} + \frac{u^2+v^2}{2\vec{c}_s^2} - \frac{u^2+v^2}{2\vec{c}_s^4} \right) \\
f_6^{eq}(i,j) &= \rho w_6 \left(1 - \frac{u}{\vec{c}_s^2} + \frac{v}{\vec{c}_s^2} - \frac{uv}{\vec{c}_s^4} + \frac{u^2+v^2}{2\vec{c}_s^2} - \frac{u^2+v^2}{2\vec{c}_s^4} \right) \\
f_7^{eq}(i,j) &= \rho w_7 \left(1 - \frac{u}{\vec{c}_s^2} - \frac{v}{\vec{c}_s^2} + \frac{uv}{\vec{c}_s^4} + \frac{u^2+v^2}{2\vec{c}_s^2} - \frac{u^2+v^2}{2\vec{c}_s^4} \right) \\
f_8^{eq}(i,j) &= \rho w_8 \left(1 + \frac{u}{\vec{c}_s^2} - \frac{v}{\vec{c}_s^2} - \frac{uv}{\vec{c}_s^4} + \frac{u^2+v^2}{2\vec{c}_s^2} - \frac{u^2+v^2}{2\vec{c}_s^4} \right).
\end{aligned} \tag{2.19}$$

2.4.6 Collision

The collision step exchanges information between the local populations and their equilibrium, and calculates the post-collision populations. There are two collision routines, one for fluid (Eq. (1.19)) and one for solid with external force density (Eq. (2.6)),

$$\begin{array}{ll} \text{Fluid Lattice} & \text{Solid Lattice} \\ f_0(i,j) = (1-\omega)f_0 + \omega f_0^{eq} & f_0(i,j) = (1-\bar{\omega})f_0 + \bar{\omega} f_0^{eq} + w_0\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_0 \\ f_1(i,j) = (1-\omega)f_1 + \omega f_1^{eq} & f_1(i,j) = (1-\bar{\omega})f_1 + \bar{\omega} f_1^{eq} + w_1\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_1 \\ f_2(i,j) = (1-\omega)f_2 + \omega f_2^{eq} & f_2(i,j) = (1-\bar{\omega})f_2 + \bar{\omega} f_2^{eq} + w_2\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_2 \\ f_3(i,j) = (1-\omega)f_3 + \omega f_3^{eq} & f_3(i,j) = (1-\bar{\omega})f_3 + \bar{\omega} f_3^{eq} + w_3\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_3 \\ f_4(i,j) = (1-\omega)f_4 + \omega f_4^{eq} & f_4(i,j) = (1-\bar{\omega})f_4 + \bar{\omega} f_4^{eq} + w_4\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_4 \\ f_5(i,j) = (1-\omega)f_5 + \omega f_5^{eq} & f_5(i,j) = (1-\bar{\omega})f_5 + \bar{\omega} f_5^{eq} + w_5\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_5 \\ f_6(i,j) = (1-\omega)f_6 + \omega f_6^{eq} & f_6(i,j) = (1-\bar{\omega})f_6 + \bar{\omega} f_6^{eq} + w_6\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_6 \\ f_7(i,j) = (1-\omega)f_7 + \omega f_7^{eq} & f_7(i,j) = (1-\bar{\omega})f_7 + \bar{\omega} f_7^{eq} + w_7\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_7 \\ f_8(i,j) = (1-\omega)f_8 + \omega f_8^{eq} & f_8(i,j) = (1-\bar{\omega})f_8 + \bar{\omega} f_8^{eq} + w_8\left(1 - \frac{\Delta t}{2\bar{\tau}}\right) F_8 \end{array} \tag{2.20}$$

with a change of variable $\bar{\omega} = \frac{1}{\bar{\tau}}$, $\bar{\tau} = \tau + \frac{\Delta t}{2}$. The force (Eq. (2.8)) on each discrete velocity space is

$$\begin{aligned} F_0 &= w_0(-u\sigma_{sx} - v\sigma_{sy}) \\ F_1 &= w_1(\frac{1-u}{c_s^2}\sigma_{sx} + \frac{-v}{c_s^2}\sigma_{sy} + \frac{u}{c_s^4}\sigma_{sx}) \\ F_2 &= w_2(\frac{-u}{c_s^2}\sigma_{sx} + \frac{1-v}{c_s^2}\sigma_{sy} + \frac{v}{c_s^4}\sigma_{sy}) \\ F_3 &= w_3(\frac{-1-u}{c_s^2}\sigma_{sx} + \frac{-v}{c_s^2}\sigma_{sy} + \frac{u}{c_s^4}\sigma_{sx}) \\ F_4 &= w_4(\frac{-u}{c_s^2}\sigma_{sx} + \frac{-1-v}{c_s^2}\sigma_{sy} + \frac{v}{c_s^4}\sigma_{sy}) \\ F_5 &= w_5(\frac{1-u}{c_s^2}\sigma_{sx} + \frac{1-v}{c_s^2}\sigma_{sy} + \frac{u+v}{c_s^4}\sigma_{sx} + \frac{u+v}{c_s^4}\sigma_{sy}) \\ F_6 &= w_6(\frac{-1-u}{c_s^2}\sigma_{sx} + \frac{1-v}{c_s^2}\sigma_{sy} + \frac{u-v}{c_s^4}\sigma_{sx} + \frac{-u+v}{c_s^4}\sigma_{sy}) \end{aligned} \tag{2.21}$$

$$F_7 = w_7(\frac{-1-u}{c_s^2}\sigma_{sx} + \frac{-1-v}{c_s^2}\sigma_{sy} + \frac{u+v}{c_s^4}\sigma_{sx} + \frac{u+v}{c_s^4}\sigma_{sy})$$
$$F_8 = w_8(\frac{1-u}{c_s^2}\sigma_{sx} + \frac{-1-v}{c_s^2}\sigma_{sy} + \frac{u-v}{c_s^4}\sigma_{sx} + \frac{-u+v}{c_s^4}\sigma_{sy})$$

where the global velocity $\vec{v} = (u, v)$ and the net external force $\vec{F} = \nabla \cdot \sigma_s = (\sigma_{sx}, \sigma_{sy})$.

2.4.7 Boundary Conditions

The boundary conditions routine refers to the boundary conditions of the simulation domain, not the solid–fluid interface. The current LB-RMT implementation makes use of the extrapolation zone to generate a smooth transition (Rycroft et al. 2018) between the solid and the fluid while preserving no-slip boundary conditions between the two phases. Contrary to the IB-LBM method (Feng & Michaelides 2004), the LB-RMT does not require any definite calls on the solid–fluid interface to interpolate the fluid velocity from the solid force density, nor to manually set the two velocities equal.

There are two types of boundary conditions involved in the simulation, periodic and no-slip. Since the simulation domain is padded with two layers of ghost nodes, they act as buffers in copying or reflecting boundary fluid nodes. The ghost nodes setup provides a relatively simple implementation (Succi 2001) on all boundary nodes, though this on-node calculation sometimes only has first-order accuracy due to the one-sided direction of population movements. More complex boundary conditions can allow second-order accuracy along the boundaries (Krüger et al. 2017).

- Periodic boundary conditions:

  On the x-axis, the left periodicity is imposed by filling the first buffer on the left with the populations of the rightmost column of the fluid, and the right periodicity is imposed by filling the first buffer on the right with the populations of the leftmost column of the fluid. The periodicities along the y-axis are analogously imposed by copying the top or the bottom row of the fluid to the respective buffer nodes (Succi 2001).

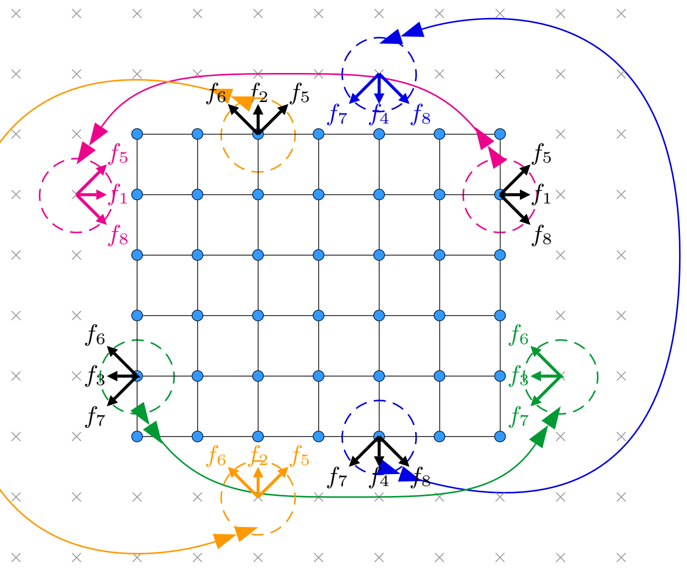

Figure 2.4: Diagram of the periodic boundary conditions on a simplified simulation domain. Along the x-axis, at the right fluid boundary, the right-outward populations (f_1, f_5, f_8) are copied to the first layer in the left buffer. And at the left fluid boundary, the left-outward populations (f_3, f_6, f_7) are also copied to the first layer in the right buffer. The buffered values in the first layer will then be streamed back to the fluid boundaries, thus imposing periodicity. Along the y-axis, the operations are analogous. At the top fluid boundary, the up-outward populations (f_2, f_5, f_6) are copied to the first layer in the bottom buffer. And at the bottom fluid boundary, the down-outward populations (f_4, f_7, f_8) are also copied to the first layer in the top buffer.

- No-slip boundary conditions:

  No-slip boundary conditions, where the fluid velocity equals the solid velocity at the boundaries, are implemented by bouncing-back of the populations leaving the fluid region. This direct on-node reflection only guarantees first-order accuracy (Succi 2001) but is quite simple to implement, just need to reflect the outward populations back by storing their values in the corresponding first-level buffer. The outward populations at each fluid boundaries are copied as the opposite direction populations in the corresponding buffer nodes. These populations then get streamed back to their original nodes, only reversed, and perform on-node bounce back. The four corners of the first buffer layer need special handling since they only have one population associated with the fluid region.

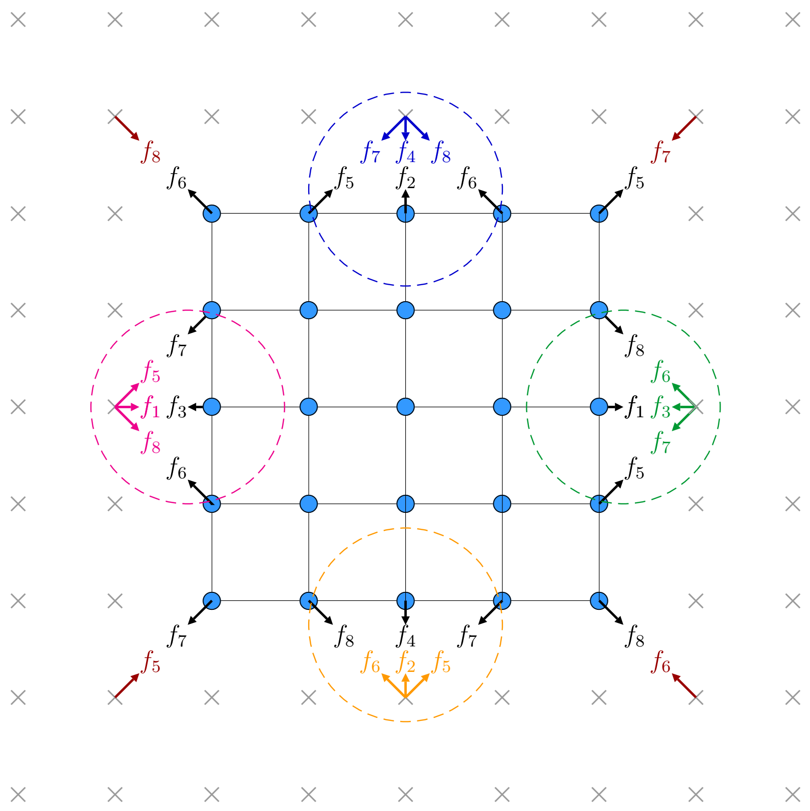

Figure 2.5: Diagram of the no-slip boundary conditions on a simplified simulation domain. For one node on the top first buffer layer, the stored f_4, f_7, f_8 populations correspond to the f_2 of its lower, f_5 of its lower-left, and f_6 of the lower-right fluid populations. For one node on the bottom first buffer layer, the stored f_2, f_5, f_6 populations correspond to the f_4 of its upper, f_7 of its upper-right, and f_8 of the upper-left fluid populations. For one node on the left first buffer layer, the stored f_1, f_5, f_8 populations correspond to the f_3 of its right, f_7 of its upper-right, and f_6 of the lower-right fluid populations. For one node on the right first buffer layer, the stored f_3, f_6, f_7 populations correspond to the f_1 of its left, f_8 of its upper-left, and f_5 of the lower-left fluid populations. The four corners in the first buffer layer are handled differently, each with only one population bouncing back to the fluid region.

Since the simulation domain is padded with two layers of buffer, they relieve the necessity to separately handling boundary conditions on the boundaries and the corners (Zou & He 1997). For the LB-native routines, only the first buffer layer is used, and the second buffer layer is solely for the second-order upwinding method in the RMT-native routines. The first-order no-slip boundary conditions is also referred as fullway bounce-back, and there are other bounce-back methods, like the halfway bounce-back (Ladd 1994), to ensure second-order accuracy on the no-slip boundary conditions.

2.4.8 Streaming

This step pulls the contribution of each f_i from the neighbouring lattices (Figure 1.3) and update the lattice of interest. It sends the $\widehat{f_i}$ values to the neighboring lattices to be used as the new f_i for the next timestep update (Eq. (1.21)). The indices represents the positions (current, left, down, right, top, lower-left, lower-right, upper-right and upper-left) of the neighboring lattices to the current lattice,

$$\begin{gathered}
f_0(i,j) = f_0(i,j) \\
f_1(i,j) = f_1(i-1,j), f_2(i,j) = f_2(i,j-1) \\
f_3(i,j) = f_3(i+1,j), f_4(i,j) = f_4(i,j+1) \\
f_5(i,j) = f_5(i-1,j-1), f_6(i,j) = f_6(i+1,j-1) \\
f_7(i,j) = f_7(i+1,j+1), f_8(i,j) = f_8(i-1,j+1).
\end{gathered} \tag{2.22}$$

2.4.9 Macroscopic Quantities Update

The original `hydro` routine (Eqs. (1.10) & (1.12)) in the LB method has been extended to calculate solid density and the global velocity (Eq. (2.23)). On a fluid node, the external force densities F_i are set to zero, allowing one uniform update rule for all macroscopic quantities. The new routine is

$$\begin{aligned}
\rho &= f_0 + f_1 + f_2 + f_3 + f_4 + f_5 + f_6 + f_7 + f_8 \\
&\quad + \frac{1}{2}(F_0 + F_1 + F_2 + F_3 + F_4 + F_5 + F_6 + F_7 + F_8) \\
u &= \frac{1}{\rho}(f_1 + f_5 + f_8 - f_3 - f_6 - f_7) + \frac{1}{2\rho}(F_1 + F_5 + F_8 - F_3 - F_6 - F_7) \\
v &= \frac{1}{\rho}(f_2 + f_5 + f_6 - f_4 - f_7 - f_8) + \frac{1}{2\rho}(F_2 + F_5 + F_6 - F_4 - F_7 - F_8).
\end{aligned} \tag{2.23}$$

The symbol ρ in Eq. (2.23) is substitute with corresponding ρ_f or ρ_s value given the label of the node.

3

Results of the Lattice-Boltzmann-Based Reference Map Technique

The LB-RMT code base is written in `C++` with an `OpemMP` multithreading support. Flow visualizations are post-processed in a `Python` plotting library `Matplotlib` with output macroscopic quantities. Three types of simulations using the LB-RMT are presented to demonstrate the accuracy, generality, and robustness of the hybrid method. First, we simulate a plane shear wave with known solutions to the reference map equations on a periodic solid domain to prove a second-order accuracy of the method. Second, we place a deformable sheet inside a periodic Poiseuille channel flow to demonstrate the feasibility and generality of the method in fluid–structure interaction. In the last simulation, we highlight the robustness of the method in handling sharp corners and large twists with an extreme case of two rotors in an initially quiescent flow. For simplicity, both the solid and the fluid are assumed to have the same density value. No explicit routines are required to specify the solid–fluid interface, nor to impose additional no-slip boundary condition along the interface.

3.1 Convergence Study of Plane Shear Waves in Solid Deformation

Stress is caused by deformation in solid, whereas it is caused by deformation rate in fluid. Since the two phases have fundamental difference in how stress is formed, it is crucial to verify whether we can extend a method that is designed for fluid mechanics to solid mechanics. This convergence study is devised to study whether the LB-RMT captures how shear waves generate solid deformation in a periodic domain $[0, L)^2$ with an initial velocity field $\vec{v}(\vec{x}, 0)$ and initial reference map field $\vec{\xi}(\vec{x}, 0)$,

$$\begin{aligned} \vec{v}(\vec{x},0) &= (0, -\omega\epsilon\sin(kx)) \\ \vec{\xi}(\vec{x},0) &= \left(x, y + \omega\epsilon\frac{\omega\cos(kx) - \lambda\sin(kx)}{\omega^2+\lambda^2}\right). \end{aligned} \tag{3.1}$$

This initial condition corresponds to a plane shear wave with the exact solutions

$$\begin{aligned} \vec{v}(\vec{x},t) &= \left(0, -e^{-\lambda t}\omega\epsilon\sin(kx-\omega t)\right) \\ \vec{\xi}(\vec{x},t) &= \left(x, y + \omega\epsilon e^{-\lambda t}\frac{\omega\cos(kx-\omega t) - \lambda\sin(kx-\omega t)}{\omega^2+\lambda^2}\right). \end{aligned} \tag{3.2}$$

where k denotes the wavenumber, ω is the angular frequency, ϵ is a small amplitude parameter, and λ is related with the energy dissipation rooted in the viscosity ν,

$$\lambda = \frac{4\pi^2}{L^2}\nu. \tag{3.3}$$

For each test, the simulation was ran for 5000 iterations with the same simulation setup: choose the length $L = 2$ m, the relaxation time $\tau_{\text{LB}} = 1.0$, the solid density $\rho_s = 1500$ kg/m^3, the shear modulus $G = 15$ Pa, the wave number $k = \frac{2\pi}{L}$, the angular frequency $\omega = \sqrt{\frac{G}{\rho_s}}$, and the small amplitude parameter $\epsilon = 0.001$ m. The first study is to change the simulation resolution $n \times n$ where $n \in [200, 400, 800, 2000]$, *i.e.* decreasing grid spacing $\Delta x \in [0.01, 0.005, 0.0025, 0.001]$ m. The simulated reference map $\vec{\xi}$ and the simulated y-component of the velocity field v of the last iteration are compared with the exact solutions. The simulation L^1 errors between the reference map field and the exact solution, as well as the velocity and the exact solution, are respectively defined as

$$\begin{gathered} E_{\vec{\xi}} = \frac{1}{n^2}\sum_i^n\sum_j^n \left|X_{\text{simulated},(i,j)} - X_{\text{exact},(i,j)}\right| + \left|Y_{\text{simulated},(i,j)} - Y_{\text{exact},(i,j)}\right|, \\ E_v = \frac{1}{n^2}\sum_i^n\sum_j^n \left|v_{\text{simulated},(i,j)} - v_{\text{exact},(i,j)}\right|. \end{gathered} \tag{3.4}$$

Figure 3.1 shows the log–log plot of the relation between the grid spacing Δx and the simulation error $E_{\vec{\xi}}$, E_v. At a relative large grid spacing ($\Delta x = 0.01$ m), the LB-RMT has error within 10^{-5}. As the grid spacing decreases, both simulation errors decrease at a similar rate. At the finest resolution of this convergence study ($\Delta x = 0.001$ m), the method has an error within 10^{-7}. Linear fitting on the log–log plot shows that both the reference map error and the velocity error are approximately of order 2. $E_{\vec{\xi}}$ is of order 1.83897 and E_v is of order 1.71657. Though a more thorough study on the grid spacing and the resolution needs to be performed, we can safely conclude that the LB-RMT is a second-order method in space, which is consistent with the parenting second-order methods.

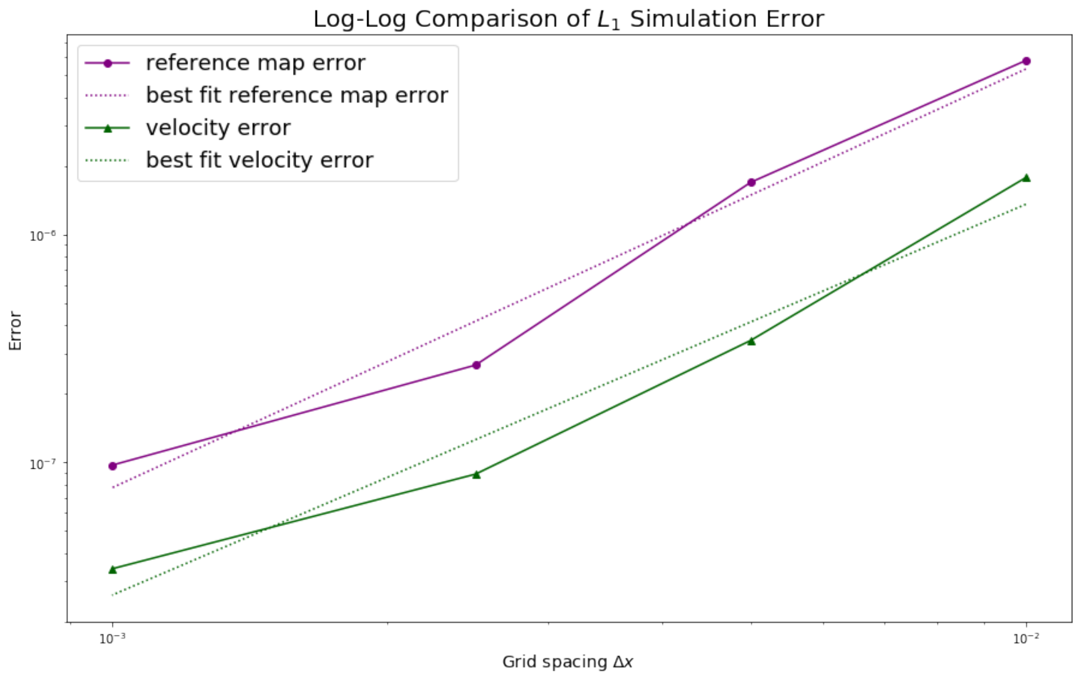

Figure 3.1: $\log$-$\log$ plot of the grid spacing Δx and the simulation error $E_{\vec{\xi}}$, E_v.

The second study is to analyze how the innate fluid kinematic viscosity ν from the fluid LB method affects the solid stress and deformation. The simulation ran at the same simulation parameters, in addition to a fixed resolution 200×200 with fixed $\Delta x = 0.01$ m. Change the relaxation time τ_{LB} in each simulation with $\tau_{\text{LB}} \in [1.0, 0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55]$. We see a very clear of increase (Figure 3.2) in the simulation errors as the relaxation time τ_{LB} decreases, *i.e.* when the viscosity ν decreases the simulation errors increase. Since the LB-RMT uses the fluid stress as the artificial viscous solid stress, and the viscosity ν, or the relaxation time τ_{LB}, controls the fluid stress, a smaller relaxation time τ_{LB} should correspond to a more accurate result, *i.e.* a smaller simulation error. This counter-intuitive relation between the relaxation time τ_{LB} and the simulation error presented in Figure 3.2 can be explained by the assumption that there is more viscosity in the system than the input value. If we modify the dissipation term λ (Eq. (3.5)) with a linear relation on the relaxation time τ,

$$\tau_m = \frac{\tau_{\text{LB}} + 1}{2}, \tag{3.5}$$

the modified $\tau_m \in [1.0, 0.975, 0.95, 0.925, 0.9, 0.875, 0.85, 0.825, 0.8, 0.775]$ is increased from the input value, *i.e.* there is more viscosity in the system. If we focus on E_v (the modification on the dissipation term does not work well with $E_{\vec{\xi}}$ given the analytical solution of $\vec{\xi}$), E_v now decreases as the relaxation time decreases, which is consistent with our argument that the simulation error decreases with the viscosity. In addition, E_v is of order 0.78626 to the relaxation time τ_{LB}. Though this value cannot provide more substantial information of the viscosity, and the linear relation we proposed is just one possible modification to the relaxation time, this modification confirms that there is more viscosity in the LB-RMT. The origin of this increase of viscosity needs to be further studied.

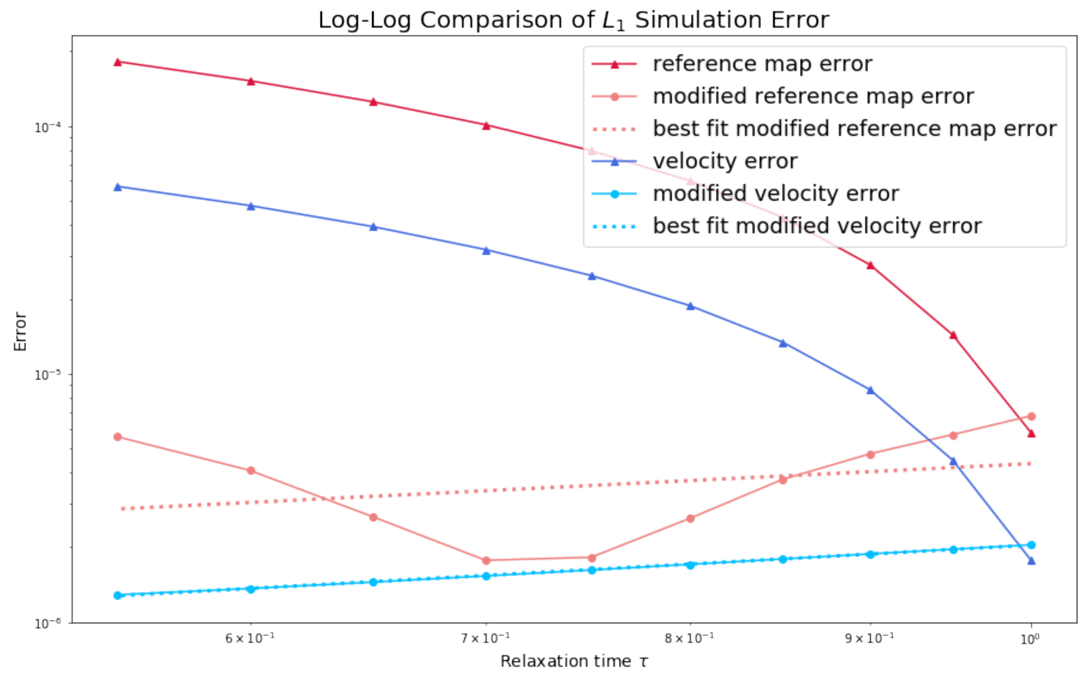

Figure 3.2: Plot of the relation between the relaxation time τ_{LB} and the simulation error $E_{\vec{\xi}}, E_v$. Modified simulation errors are plotted in dashed lines. Though the simulations are more accurate with the modification in the input viscosity, the $\log$-$\log$ plot of the modified simulation errors does not conclude more information about their convergence.

3.2 Deformable Sheet in the Poiseuille Flow

The second LB-RMT example is a simple fluid–structure interaction that involves only one deformable object in a periodic channel flow. A deformable sheet of size 0.2×0.3 m^2 with an identity reference map is centered at $(0.6\text{ m}, 0.5\text{ m})$, where a uniform force density that resembles a pressure-driven Poiseuille flow is applied to all fluid nodes. The Reynolds number and the relaxation time of the fluid are set to $Re = 10$ and $\tau_{\text{LB}} = 1.0$, leading to the maximum velocity value of a purely Poiseuille flow to be approximately 0.0167 m/s. The solid has small-strain modulus $G = 0.15$ Pa, meaning the sheet can easily undergo large deformations. Both the fluid and the solid have density 1500 kg/m^3. The simulation domain is $[0, 3) \times [0, 1)$ m^2, with 300×100 resolution and a physical grid spacing $\Delta x = 0.01$ m. The simulation ran for 25000 iterations with a physical timestep $\Delta t \approx 0.577$ s (Appendix A) and outputted every 50 frames. Supplemental Movie 1 and 2 show the complete simulation.

The hybrid method successfully simulates a Poiseuille channel flow, and the sheet deforms with the flow while moving to the end of the channel. The solid reference map is represented by the dashed mesh, whose distortion shows how much the sheet has been deformed. As the simulation time increases, the sheet starts from a uniform rectangle, unmoved and undeformed, to a similar-boomerang shape, stretched and squished. This shape is consistent with the parabolic velocity profile of the Poiseuille flow. The fluid at the center of the channel has the maximum velocity, thus pushing the center part of the sheet forward faster compared with top and bottom. Due to the velocity difference surrounding the solid, the sheet becomes more stretched at the lower-left and upper-left corners. After the flow has been fully developed, the sheet continues to move to the end of the channel without more changes in the reference map. Though the level set of the solid–fluid interface becomes discontinuous at the two corners of the sheet near the end of the simulation due to the limitation of the grid spacing, the hybrid method has proven to be stable even with a relatively large grid spacing value.

The density field of the solid and the fluid (Figure 3.3) changes with their interaction. If we were only having a Poiseuille flow with no solid immersed, the density field of the fluid should remain unchanged due to the conservation of mass. However, the existence of the compressible solid sheet changes the density profile of the fluid. Since the flow is pushing the sheet forward, more fluid clusters behind the solid. As the simulation time increases, the fluid density increases at the concave boundary of the solid, causing a decrease of the fluid density at the solid front. Inside the solid sheet, however, the density remains mostly unchanged, except at the front and back boundaries. Since the constitutive relation applied to the solid is compressible, the increase of the solid density at the concave side is consistent with inertia. The solid particles would resist the change and hope to remain at their original positions, causing a delay in their particle movement. Though there is no special handling to the solid density and the fluid density, nor the solid–fluid interface, the hybrid method is capable of distinguishing the two phases and generating a clear density division between them.

The global velocity field (Figure 3.4) is consistent with a Poiseuille channel flow. The velocity starts out stationary, and as simulation time increases, the fluid flow is fully developed to a parabolic velocity profile with the maximum velocity along the center axis. The fluid is capable of moving the solid forward, whereas the solid is also capable of parting the fluid. The second phenomenon becomes more visible after $t = 10000\Delta t$, where the velocity profile near the wall boundaries is bent because the deformed sheet alters the path where the fluid can pass. Snapshot at $t = 24500\Delta t$ shows that the fluid velocity decreases at the concave boundary of the solid, which is consistent with the physical intuitive that when a fluid flow knows there is a solid in front of it, it will find another path and have lower velocity near the solid boundary. The solid also has a lower velocity inside compared with the fluid flow. This means the solid initially moves with the flow speed, then it decreases. This loss of kinematic energy is likely to be caused by viscosity, or due to the solid is compressible. The fluid flow also has not reached the maximum value for a purely Poiseuille flow, which is likely to be caused by the existence of a deformable solid and their interaction. Overall the LB-RMT shows a very smooth velocity field transition and velocity update between the two phases.

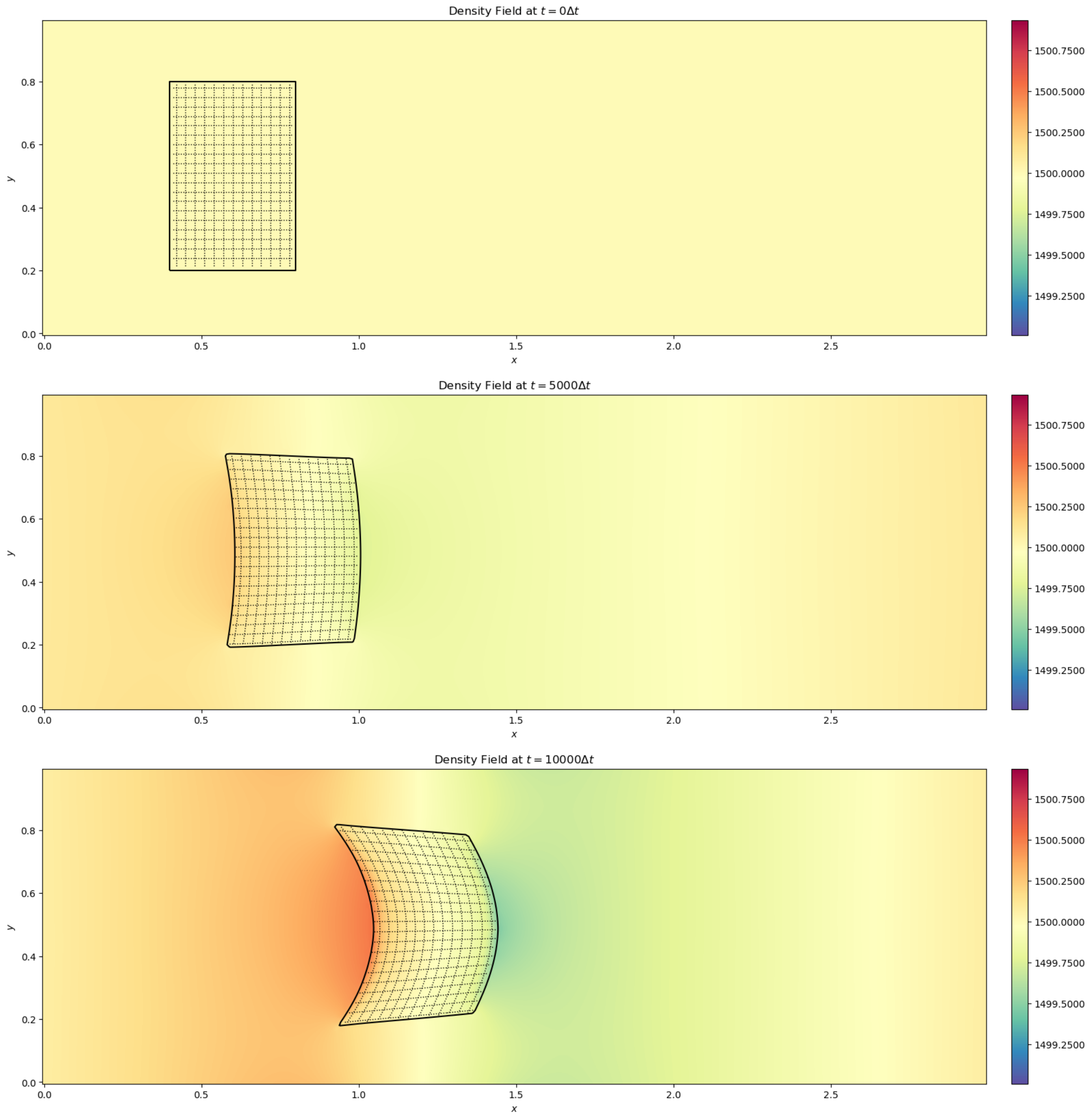

Figure 3.3: Snapshots at $t = 0, 5000\Delta t, 10000\Delta t$ of the density fields of the channel flow and the deformable sheet. Density undergoes very little change in the first half of the simulation, and is conserved in most regions.

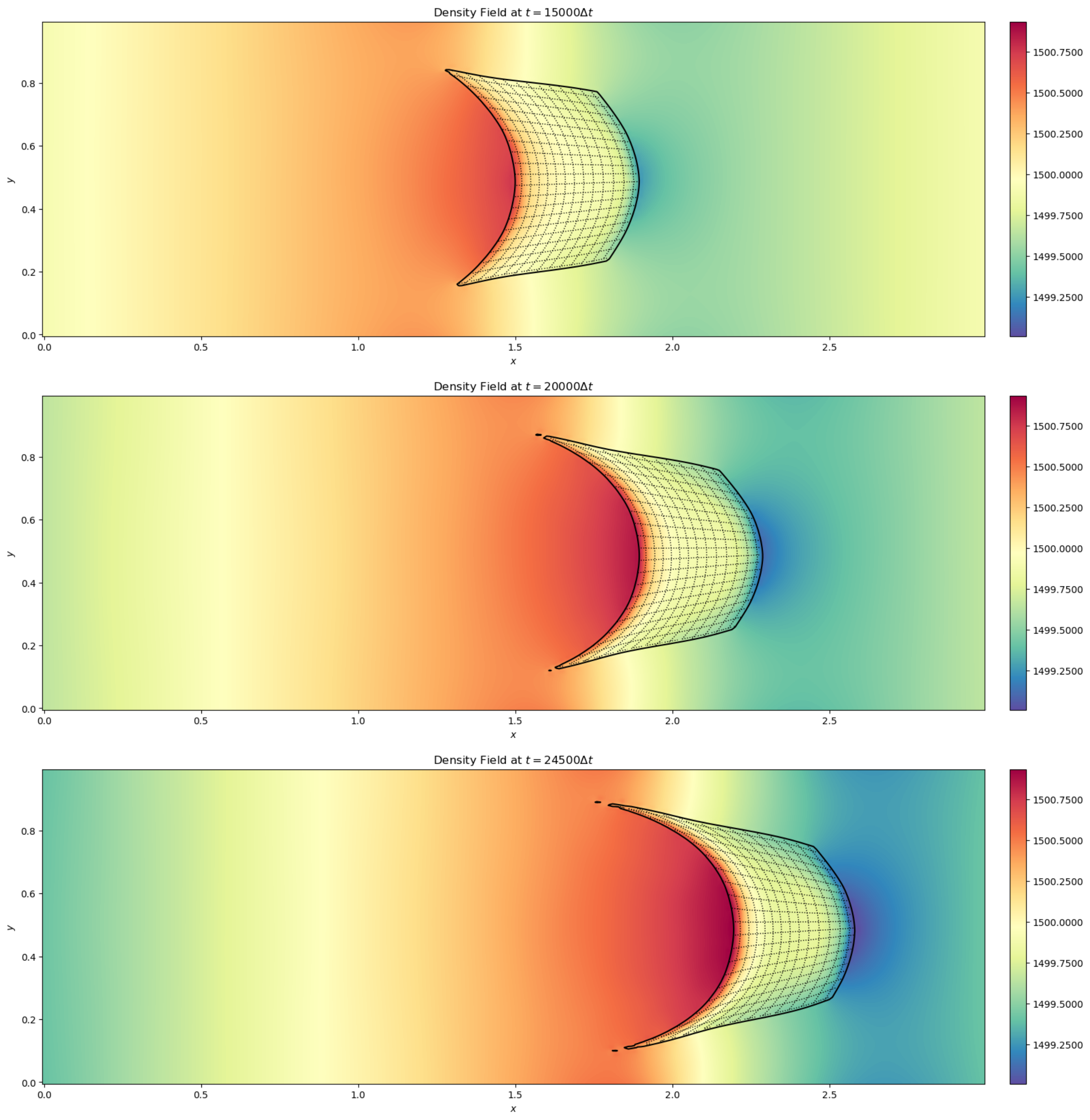

Figure 3.3: Snapshots at $t = 15000\Delta t, 20000\Delta t, 24500\Delta t$ of the density fields of the channel flow and the deformable sheet. Density changes are more prominent near the solid boundary when the solid is more deformed.

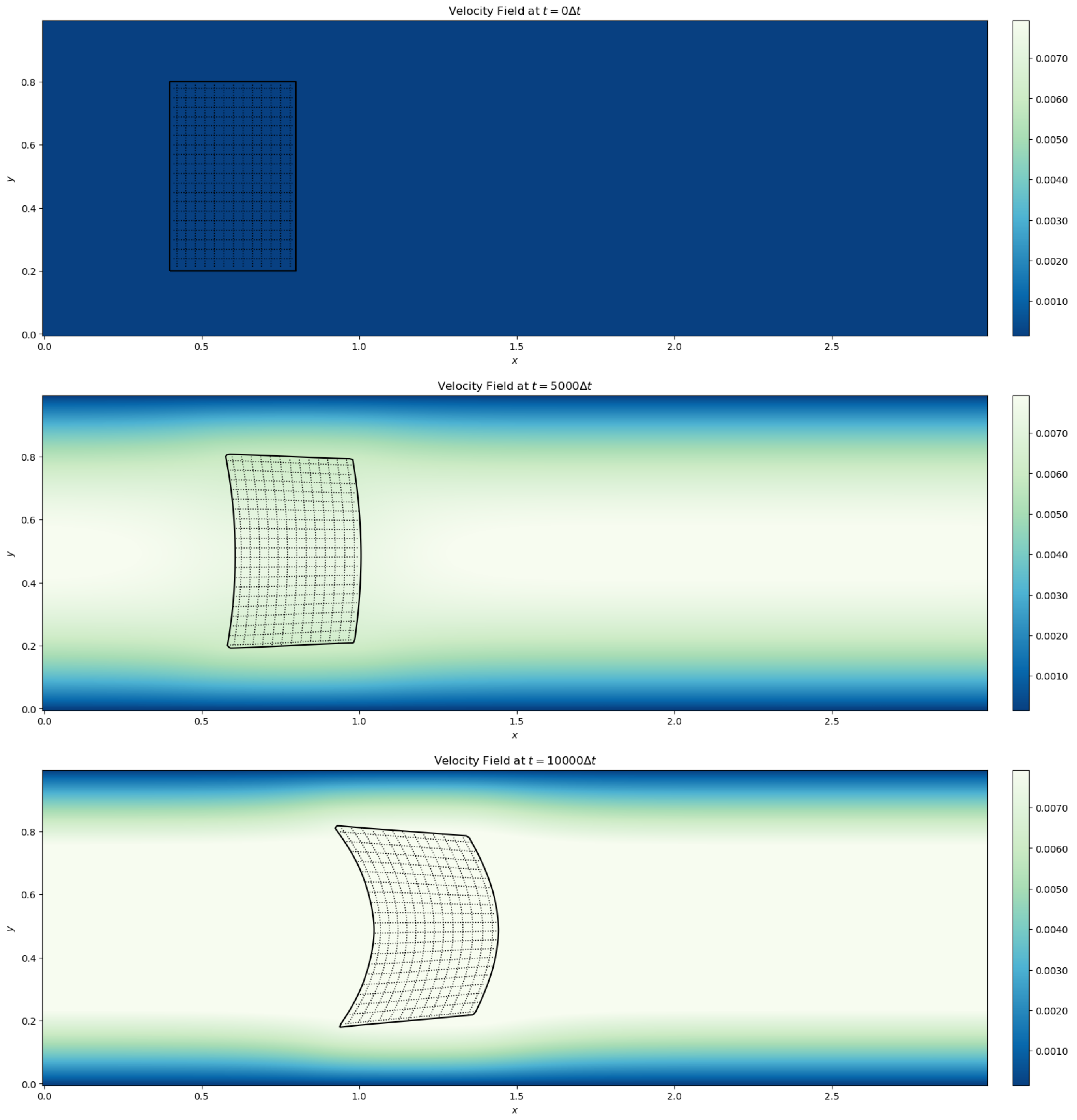

Figure 3.4: Snapshots at $t = 0, 5000\Delta t, 10000\Delta t$ of the velocity field. A parabolic velocity profile of the Poiseuille flow develops along the channel. The fluid moves the sheet forward.

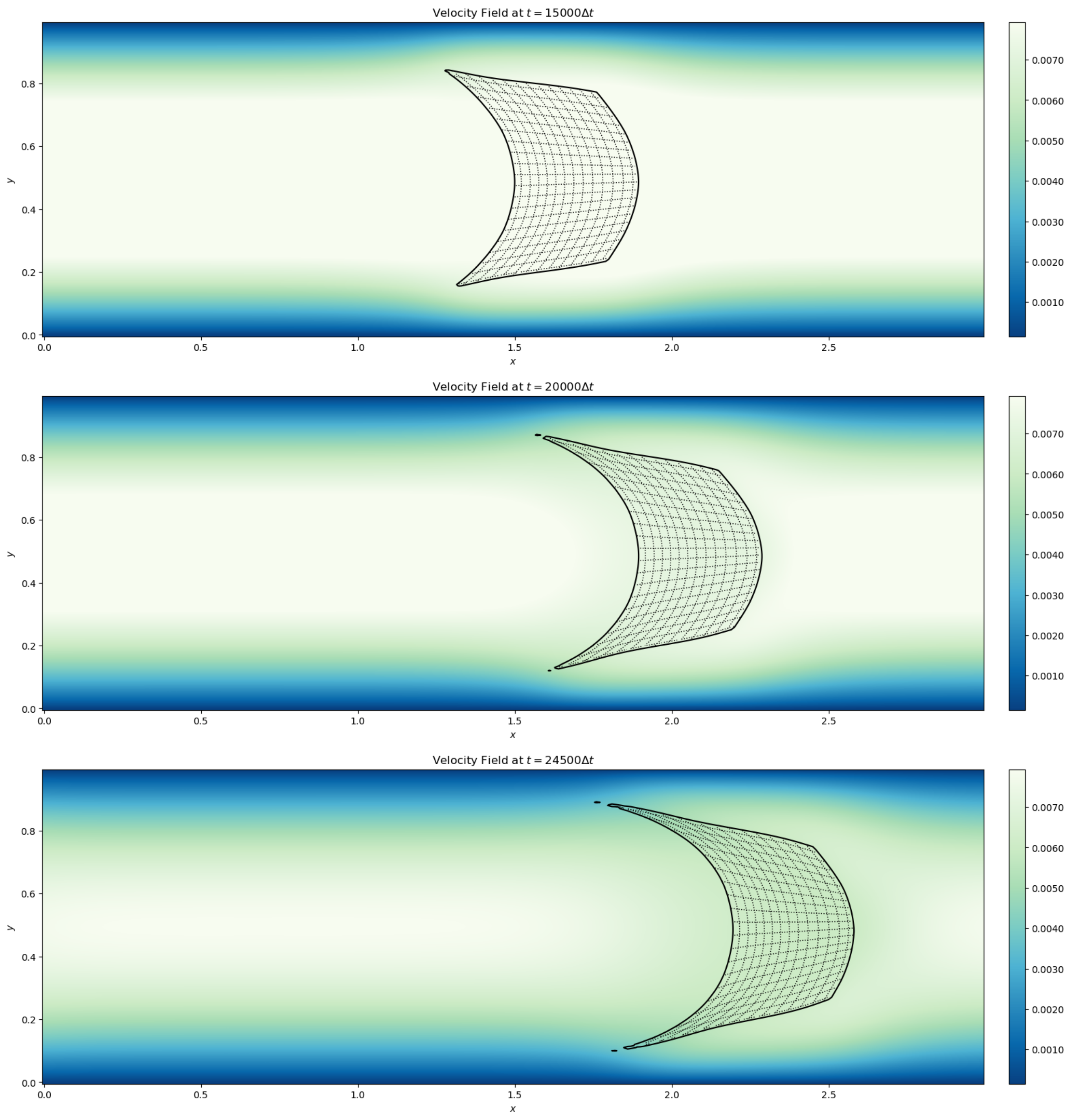

Figure 3.4: Snapshots at $t = 15000\Delta t, 20000\Delta t, 24500\Delta t$ of the velocity field. The stretched sheet has lower velocity inside the solid, and changes the standard Poiseuille flow velocity profile near the no-slip wall boundaries.

3.3 Rotors in Initially Quiescent Flow

A rotor is placed in the center of an initially quiescent periodic channel flow. A periodic rotational force is applied in the center circular region of the rotor. This anchoring force rotates the rotor first clockwise then counter-clockwise. The rotor spins and twists with the anchor force, causing the initially stationary flow to move as the simulation time increases. This example is modeled after the seven-pointed rotor presented by Rycroft et al. (2018) to test the stability and limitation of the LB-RMT in capturing the solid–fluid interface of extreme cases like sharp corners and large twists.

The rotor is centered at $(1.5 \text{ m}, 0.5 \text{ m})$ and has a vertex at $\big(0.62\cos\big(\frac{2\pi k}{7}\big), \sin\big(\frac{2\pi k}{7}\big)\big)$ for $k \in \mathbb{Z}$. Both the fluid and the rotor have density 1500 kg/m^3 and are set to be stationary at the start of the simulation. The center region of $r < 0.02$ m is pivoted with an oscillatory anchor force $\vec{f}(\vec{x}) = -0.00001(|\vec{x}| - 0.16)(R_{\theta(t)}\vec{\xi}(\vec{x}))$ where $R_{\theta(t)}$ is the rotational matrix with angle $\theta(t) = \pi(1 - \cos(t))$. The domain is the same as the previous example, except the solid small-strain modulus is now $G = 1$ Pa. The simulation ran for 8000 iterations and outputted every 20 frames. It broke after the 4000-th iteration due to reference map extrapolations. The vorticity field (Figure 3.5) shows the rotor deforms first clockwise then counter-clockwise with the oscillatory angular acceleration. Vortices are shed along the solid–fluid interface and excite the fluid flow. The LB-RMT preserves the sharp corners across the simulation, and can hold large twists and bends to the rotor periodically. Supplemental Movie 3 shows the complete simulation, where the seven-hold symmetry is preserved.

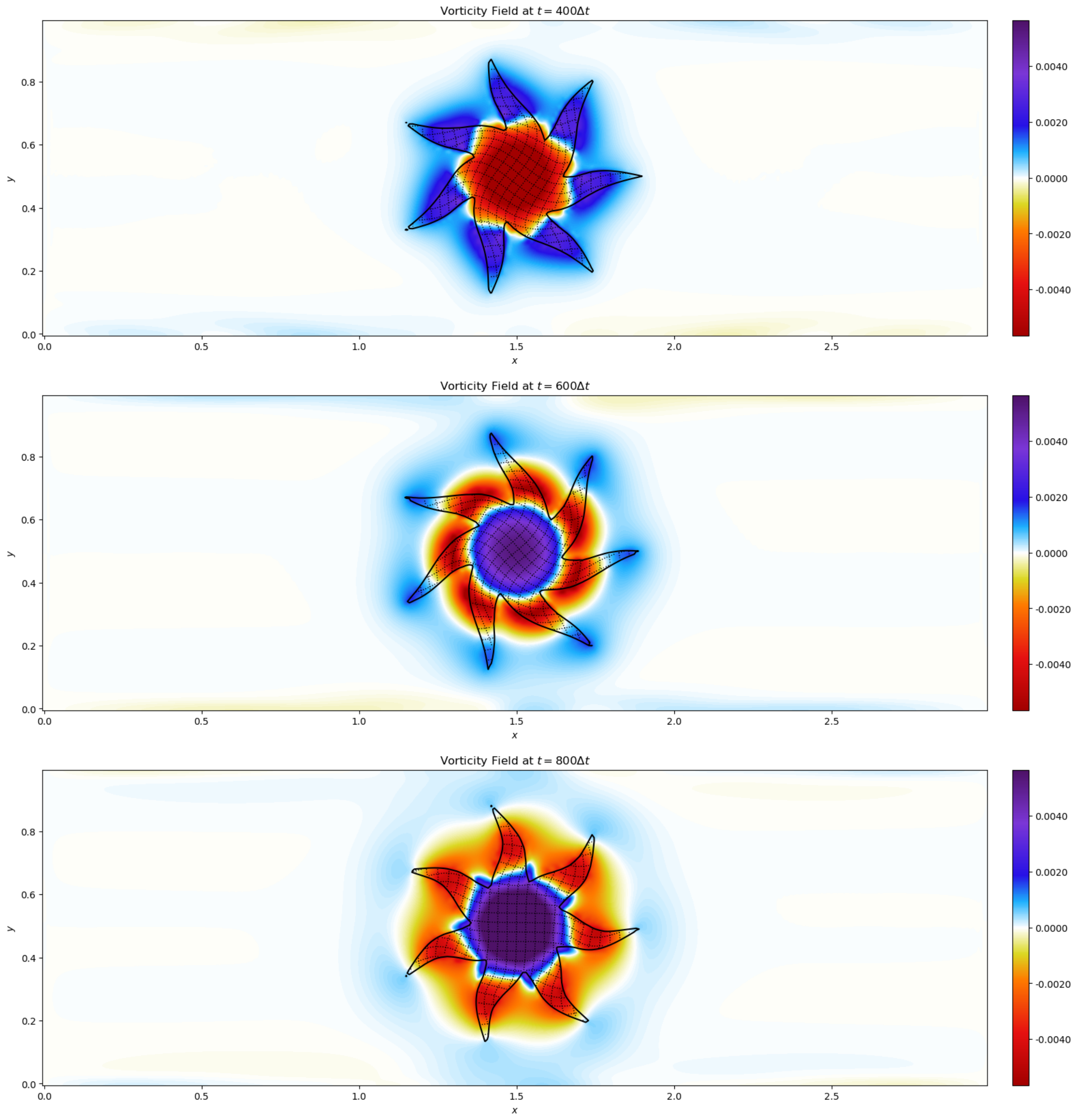

Figure 3.5: Snapshots at $t = 400\Delta t, 600\Delta t, 800\Delta t$ of the vorticity field of one rotor. Rotor spins from clockwise to counter-clockwise. Vortices are shed from the solid–fluid interface, and the fluid near the rotor has been excited.

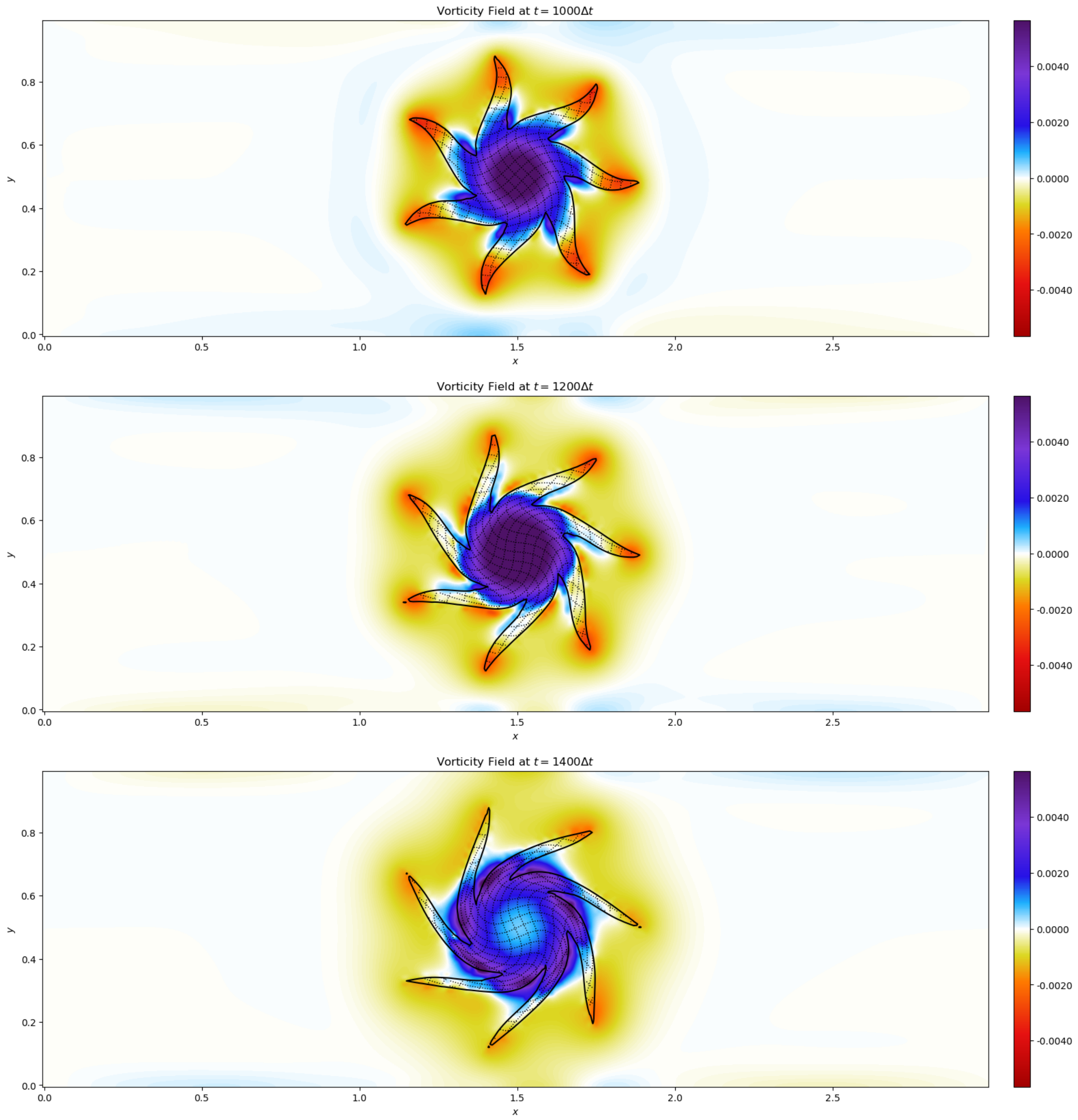

Figure 3.5: Snapshots at $t = 1000\Delta t, 1200\Delta t, 1400\Delta t$ of the vorticity field of one rotor. The rotor continues to spin counter-clockwise, and is very twisted at $t = 1400\Delta t$. More fluid has been excited due to the rotor rotation.

A second rotor is then added to the channel to introduce more variations. The two rotors are now center at $(0.75 \text{ m}, 0.5 \text{ m})$ and $(2.25 \text{ m}, 0.5 \text{ m})$, with new anchor forces of $\vec{f}(\vec{x}) = \pm 0.000025(|\vec{x}| - 0.16)(R_{\theta(t)}\vec{\xi}(\vec{x}))$. The simulation ran for 50000 iterations and outputted every 100 frames. The two rotors spin at opposite directions with opposite oscillatory anchor forces. Supplemental Movie 4 shows the complete simulation, where more disturbance happens to the left rotor.

The vorticity field (Figure 3.6) first shows the two rotors still preserve the seven-pointed symmetry. But as the flow exits from the right and re-enters from the left due to the periodicity along the x-axis, the fluid breaks the symmetry of the left rotor, whose rotation starts to be asymmetrical and unpredictable. Since the anchor force is still being applied to the left rotor, it slips around the anchoring region and presents a swimmer-like movement. However, the left rotor still tries to restore to its original center position as the simulation time increases. The simulation broke after the 46500-th iteration; by then both rotors have been stretched and become asymmetrical.

The snapshots of the vorticity field in Figure 3.6 and Figure 3.7 capture the different states of the spinning rotors. Figure 3.6 shows the steady state of the two rotors, when both rotors still have symmetry. The vortices are being shed from the solid–fluid interface, and the two rotors have the opposite vorticity fields. Once the symmetry is lost, the motion of the rotors becomes unpredictable. Figure 3.7 shows the left rotor slips around the anchor and loses its symmetry with the swimmer-like movement. However, the LB-RMT is still capable of simulating the rotors, preserving the sharp corners across the simulation, and tracking the solid–fluid interface in these extreme periodic rotational cases.

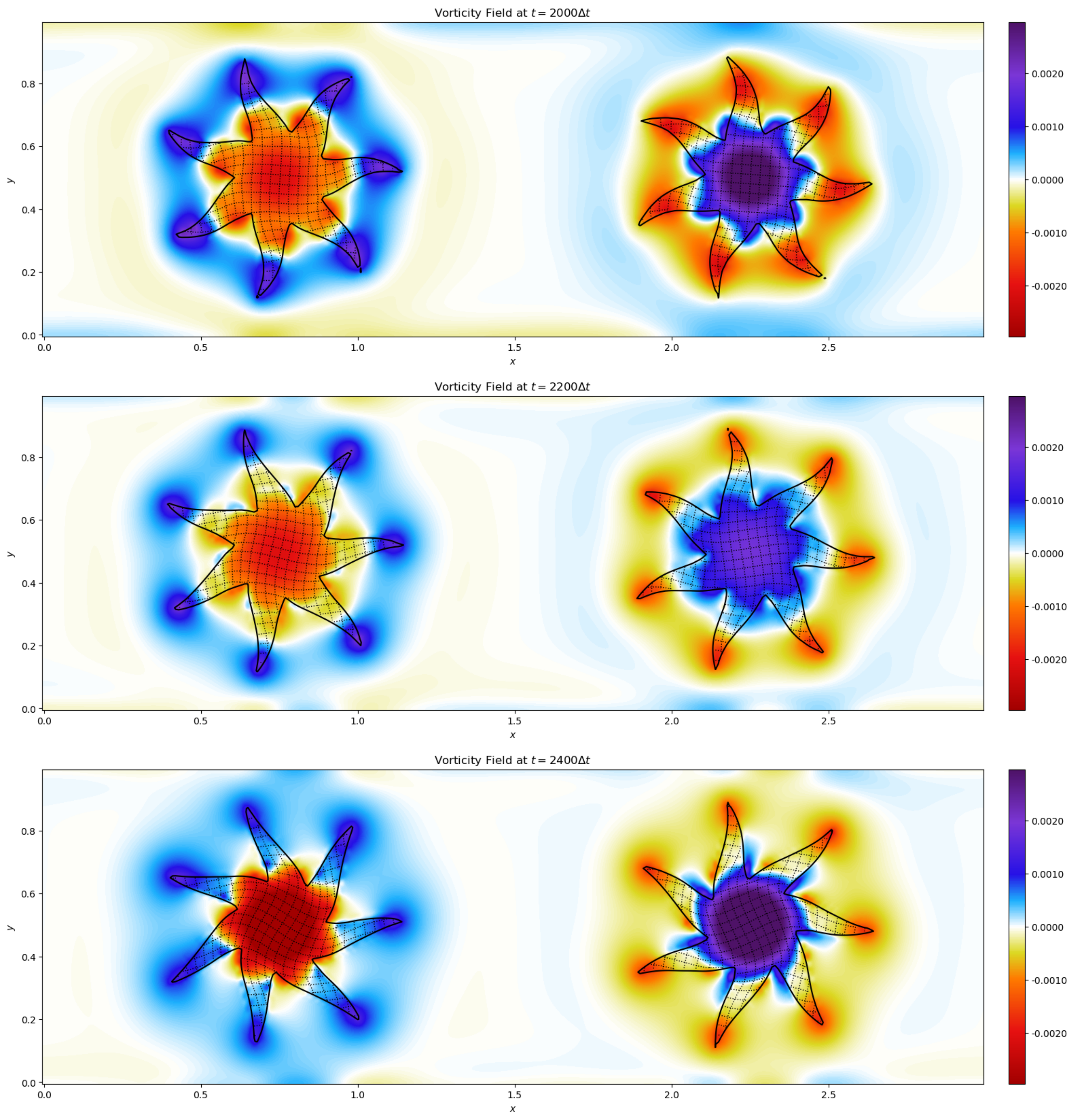

Figure 3.6: Snapshots at $t = 2000\Delta t, 2200\Delta t, 2400\Delta t$ of the vorticity field of two rotors. The left rotor spins clockwise and the right counter-clockwise. Opposite vortices are shed from the seven points of the rotors.

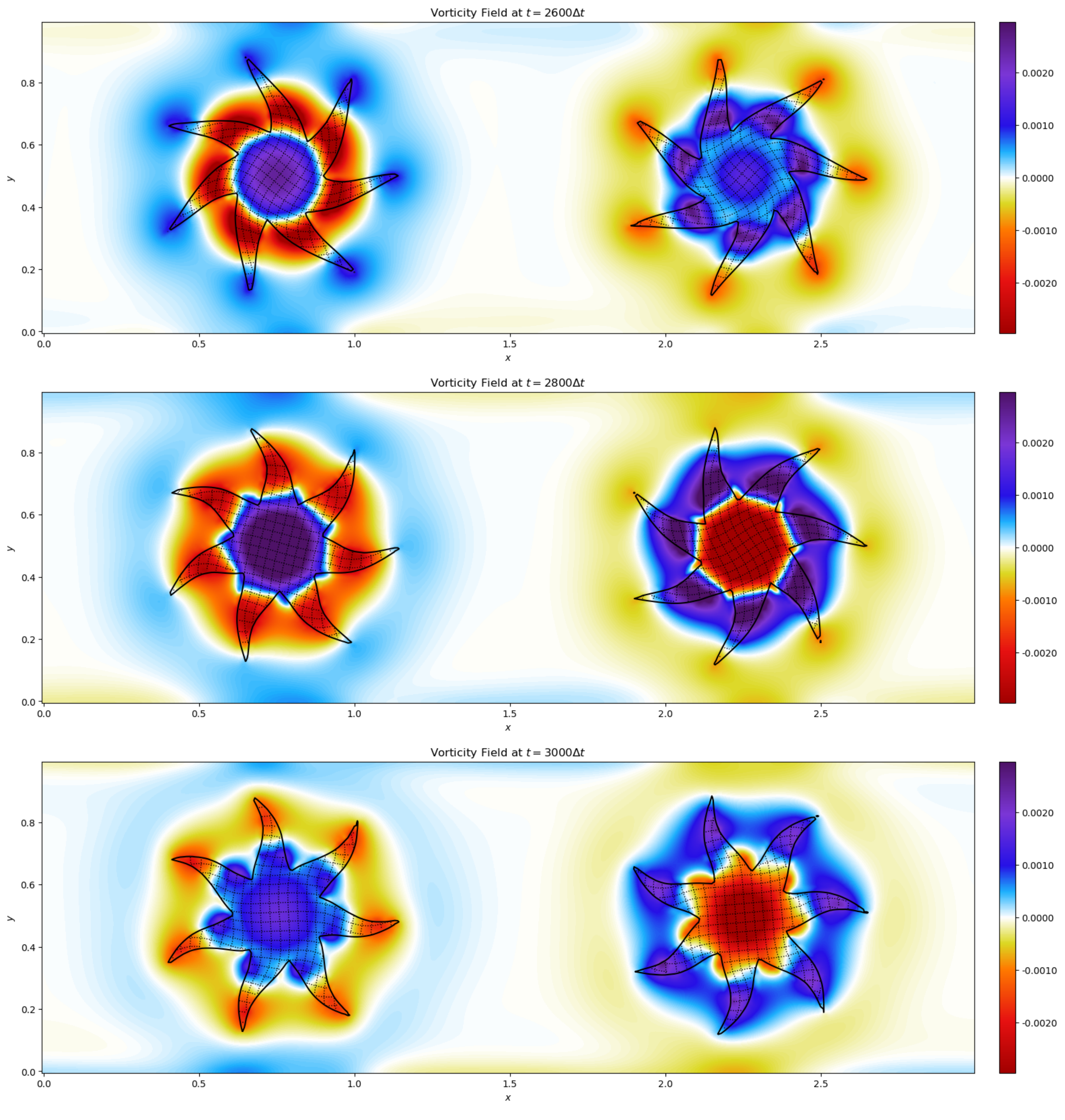

Figure 3.6: Snapshots at $t = 2600\Delta t, 2800\Delta t, 3000\Delta t$ of the vorticity field of two rotors. The left rotor switches spinning directions to counter-clockwise and the right to clockwise. The entire channel flow has been excited.

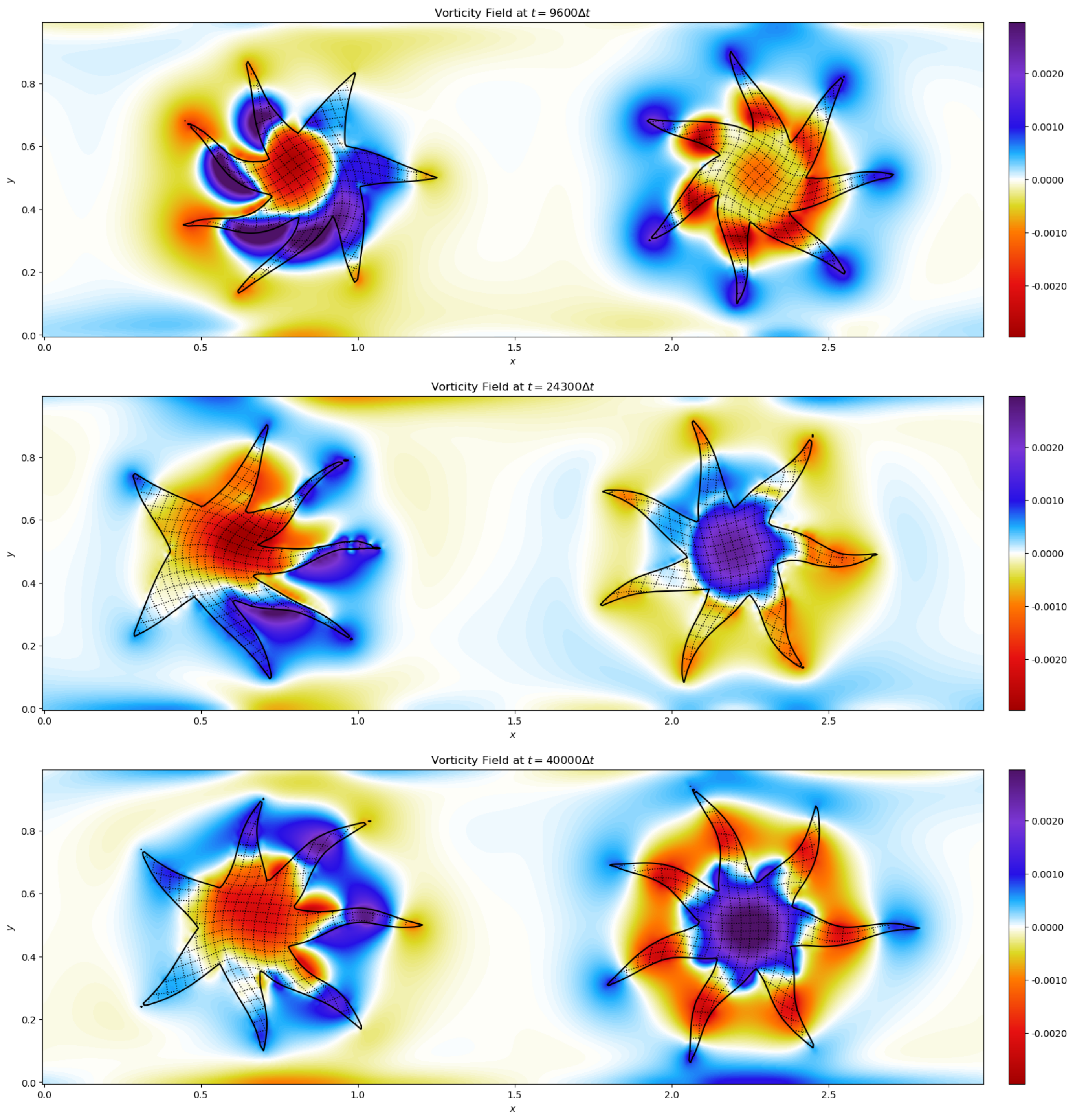

Figure 3.7: Snapshots at $t = 9600\Delta t, 24300\Delta t, 40000\Delta t$ of the vorticity field of two rotors. The left rotor experiences more distortion due to flow disturbance, where some of its angles have swimmer-like movements.

4 Conclusion

The thesis combines two Eulerian-based methods (Rycroft et al. 2018; Succi 2001) together to simulate fluid–structure interactions. The resulting hybrid lattice-Boltzmann-based reference map technique (LB-RMT) combines the benefits of the two methods: the reference map technique provides a straightforward Eulerian description of large solid deformation, and the lattice Boltzmann method with a simple implementation of fluid simulation. The LB-RMT formulates a mesoscopic representation of solid deformation, and provides a novel approach to combining the fluid and the solid phases.

It is capable of generating accurate simulation of solid deformation, fluid motion and their interaction, also has impressive results in handling large twists and sharp corners of flexible solids in fluid flow.

There are number of extensions to the numerical procedure to consider. First, the hybrid method needs to change the no-slip boundary conditions implementation from a first-order on-node (or full-way) bounce-back method to a second-order halfway bounce-back (Krüger et al. 2017). The order of the LB method is determined by that of its boundary conditions. Since all the other update rules and discretization stencils are second-order in the current implementation, the LB-RMT should have the potential to preserve second-order accuracy in more generalized simulation domain in addition to fully periodic. Second, the method should remove the restriction on the same density requirement of the two phases. Though it is quite common to have the same solid and fluid densities in some biological applications, this restriction limits the hybrid method from being suitable to more applications with varying densities between the phases, or even within a single phase. One possible solution is to extend the halfway bounce-back boundary conditions along the solid–fluid interface. This supplement also opens the discussion of flexible solid boundary conditions for fluid–structure interaction in the lattice Boltzmann literature. Lastly, the current code base has not fully optimized the parallelization feature of the LB method. The nested-loops are not scaled properly with the number of threads to perform multithreading, which need to be more carefully planned.

The results of the LB-RMT demonstrate the compatibility of the reference map technique with general fluid simulation methods and the feasibility of a uniform mesoscopic framework for both the fluid and the solid. It underpins a base structure of numerical schemes with many potential future directions. The solid–fluid interface study can be extended to handle dissolving solids with different densities, which will be useful in fluid–sediment mixture and erosion simulations. In addition, the LB-RMT can increase the simulation phases from two to three. Since the lattice Boltzmann method is well developed in fluid–gas simulation (Shan & Chen 1993; Swift et al. 1996), the hybrid method can be further expanded to simulate multispecies cases like fluid–solid–gas simulation in both two and three dimensions. It will unify the mesoscopic description for the three phases of matter and encompass the different discretization setups and numerical schemes into one method. This overarching LB-RMT can be combined with additional conditions, like chemical reactions or thermodynamics changes, to create a time-lapse simulation of object phase change and multi-object reaction. One of the many aspirations of the LB-RMT is to develop a production-level fluid–structure interaction solver for physics-based animation. The LB method has been used to produce fluid animation in game industry (Judice et al. 2010), as well as some LB-based fluid simulation packages in standard 3D computer graphics software like `Blender` (Thurey 2007). Its lightweight in computation and flexibility in handling complex geometry make it suitable for physics-based simulation, especially for real-time game animation and rendering. Throughout the development of graphic cards and rendering techniques, the physics-based simulation in animation and game industry has always been seeking the balance among performance, accuracy and aesthetics. The LB-RMT will readily provide a state-of-the-art solution to generate solid, fluid, gas and their interactions with only one method needed.

A

Units Conversion

The convention in the lattice Boltzmann (LB) method is to set the discrete timestep Δt and the discrete lattice spacing Δx to be unity and dimensionless,

$$\Delta t = 1 \tag{A.1}$$

$$\Delta x = 1. \tag{A.2}$$

Then the lattice speed c_{i} along the axis is also unity and dimensionless,

$$c_{\mathrm{i}} = \frac{\Delta x}{\Delta t} = 1. \tag{A.3}$$

The speed of sound in the LB method is chosen to be $c_{\mathrm{s}} = \sqrt{\frac{1}{3}}$, which is a dimensionless constant and can be derived by the relation

$$c_{\mathrm{s}}^2 = \sum_i w_i c_i^2 = \frac{1}{3} \text{ (for most lattices).} \tag{A.4}$$

Since ones from the unities are hard to keep track of in the upcoming units conversion, let us first introduce some dimensionless units in the LB method. The previous Δt is actually Δt_{LBM}, and similarly Δx is Δx_{LBM}. Let $\mathcal{T}$ denote time and $\mathcal{L}$ length, then

$$\Delta t_{\mathrm{LBM}} = \mathcal{T} \tag{A.5}$$

$$\Delta x_{\mathrm{LBM}} = \mathcal{L} \tag{A.6}$$

$$c_{i,\mathrm{LBM}} = \frac{\mathcal{L}}{\mathcal{T}} \tag{A.7}$$

$$c_{s,\mathrm{LBM}} = \sqrt{\frac{1}{3}} \frac{\mathcal{L}}{\mathcal{T}}. \tag{A.8}$$

A conversion between the physical value of the speed of sound and the LB speed of sound is

$$c_{s,\mathrm{ph}} = c_{s,\mathrm{LBM}} \frac{\Delta x_{\mathrm{ph}}}{\Delta t_{\mathrm{ph}}}. \tag{A.9}$$

The relation above gives the conversion between the LB units and the physical units. For example, the compressible solid has a density of $\rho_{\mathrm{s}} = 1500$ kg/m^3, and the shear modulus $G = 15$ kg/(m·s^2) (or

Pa). The compressible solid has the speed of sound $c_{\text{s,ph}}$, *i.e.* the shear wave speed,

$$c_{s,\text{ph}} = \sqrt{\frac{G_{\text{ph}}}{\rho_{s,\text{ph}}}} = 0.1 \text{ m/s} \tag{A.10}$$

where the subscript $_{\text{ph}}$ is added to distinguish from the LB quantities. Suppose $\Delta x_{\text{ph}} = 0.01$ m, with this known physical speed of sound $c_{\text{s,ph}}$, one LB timestep corresponds to one physical timestep

$$\Delta t_{\text{ph}} = \frac{c_{s,\text{LBM}}}{c_{s,\text{ph}}} \Delta x_{\text{ph}} = \frac{\sqrt{\frac{1}{3}}}{0.1 \text{ m/s}} \times 0.01 \text{ m} = \frac{\sqrt{3}}{30} \text{ s}. \tag{A.11}$$

Therefore, the key to connect the LB units and the physical units is the choice of Δx_{ph}. Once the simulation has set a value for Δx_{ph}, the physical timestep Δt_{ph} can be calculated. With the conversion between time and space, other LB and physical quantities can be converted accordingly. Take the shear modulus as an example, it has a dimensional unit of $\frac{\text{mass}}{\text{length}\cdot\text{time}^2}$. Consider unit mass $\mathcal{M}$, then a conversion between the physical value of the shear modulus and the LB shear modulus is

$$G_{\text{ph}} = G_{\text{LBM}} \frac{\mathcal{M}}{\Delta x_{\text{ph}} \cdot \Delta t_{\text{ph}}^2}. \tag{A.12}$$

Since Δx_{ph} is by choice and Δt_{ph} has been calculated given the physical values,

$$G_{\text{LBM}} = G_{\text{ph}} \cdot \Delta x_{\text{ph}} \cdot \Delta t_{\text{ph}}^2 \cdot \frac{1}{\mathcal{M}} = 15 \cdot \mathcal{M}/(\text{m}\cdot\text{s}^2) \cdot 0.01 \text{ m} \cdot (\frac{\sqrt{3}}{30} \text{ s})^2 \cdot \frac{1}{\mathcal{M}} = \frac{1}{2000}. \tag{A.13}$$

Other simulation parameters, like the density ρ and kinematic viscosity ν, can be converted between the LB units and the physical units analogously,

$$\rho_{\text{ph}} = \rho_{\text{LBM}} \Delta x_{\text{ph}}^3 \Leftrightarrow \rho_{\text{LBM}} = \rho_{\text{ph}} \frac{1}{\Delta x_{\text{ph}}^3} \tag{A.14}$$

$$\nu_{\text{ph}} = \nu_{\text{LBM}} \frac{\Delta x_{\text{ph}}^2}{\Delta t_{\text{ph}}} \Leftrightarrow \nu_{\text{LBM}} = \nu_{\text{ph}} \frac{\Delta t_{\text{ph}}}{\Delta x_{\text{ph}}^2}. \tag{A.15}$$

The following table shows the values used of simulation parameters in the LB method, and their corresponding physical values. Boxed values are preset or given during the simulation. Note that all LB values are dimensionless.

Simulation Parameter	LB Value	Physical Value
Timestep Δt	$\boxed{1}$	$\sqrt{3}/30$ s
Lattice Spacing Δx	$\boxed{1}$	$\boxed{0.01 \text{ m}}$
Speed of Sound c_s	$\boxed{\sqrt{1/3}}$	0.1 m/s
Solid Density ρ_s	$3/2000$	$\boxed{1500 \text{ kg/m}^3}$
Shear Modulus G	$1/2000$	$\boxed{15 \text{ kg/(m} \cdot \text{s}^2)}$

Table A.1: Value conversions between the LB units and the physical units.

References

Aidun, C. K. & Clausen, J. R. (2010). Lattice-Boltzmann method for complex flows. *Annual Review of Fluid Mechanics*, 42, 439–472.

Almgren, A. S., Bell, J. B., & Szymczak, W. G. (1996). A numerical method for the incompressible Navier–Stokes equations based on an approximate projection. *SIAM Journal on Scientific Computing*, 17(2), 358–369.

Amati, G., Succi, S., & Piva, R. (1997). Massively parallel lattice-Boltzmann simulation of turbulent channel flow. *International Journal of Modern Physics C*, 8(04), 869–877.

Anderson, D., Tannehill, J. C., & Pletcher, R. H. (1997). *Computational fluid mechanics and heat transfer*. Taylor & Francis.

Aslam, T. D. (2004). A partial differential equation approach to multidimensional extrapolation. *Journal of Computational Physics*, 193(1), 349–355.

Bagchi, P. (2007). Mesoscale simulation of blood flow in small vessels. *Biophysical Journal*, 92(6), 1858–1877.

Baliga, B. & Patankar, S. (1980). A new finite-element formulation for convection-diffusion problems. *Numerical Heat Transfer*, 3(4), 393–409.

Batchelor, G. (2000). *An Introduction to Fluid Dynamics*. Cambridge University Press.

Bathe, K.-J. (2006). *Finite element procedures*. Klaus-Jurgen Bathe.

Bazilevs, Y., Calo, V. M., Zhang, Y., & Hughes, T. J. (2006). Isogeometric fluid–structure interaction analysis with applications to arterial blood flow. *Computational Mechanics*, 38(4-5), 310–322.

Bazilevs, Y., Takizawa, K., & Tezduyar, T. E. (2013). *Computational fluid–structure interaction: methods and applications*. John Wiley & Sons.

Bell, J. B., Colella, P., & Glaz, H. M. (1989). A second-order projection method for the incompressible Navier–Stokes equations. *Journal of Computational Physics*, 85(2), 257–283.

Belytschko, T., Liu, W. K., Moran, B., & Elkhodary, K. (2013). *Nonlinear finite elements for continua and structures*. John Wiley & Sons.

Bernaschi, M., Melchionna, S., Succi, S., Fyta, M., Kaxiras, E., & Sircar, J. K. (2009). MUPHY: A parallel multi physics/scale code for high performance bio-fluidic simulations. *Computer Physics Communications*, 180(9), 1495–1502.

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*, 94(3), 511.

Briggs, W. L., Henson, V. E., & McCormick, S. F. (2000). *A Multigrid Tutorial: Second Edition*. SIAM.

Chapman, S. & Cowling, T. G. (1952). *The mathematical theory of non-uniform gases*. Cambridge University Press.

Chen, S. & Doolen, G. D. (1998). Lattice Boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, 30(1), 329–364.

Chorin, A. J. (1967). A numerical method for solving incompressible viscous flow problems. *Journal of Computational Physics*, 2(1), 12–26.

Chorin, A. J. (1968). Numerical solution of the Navier–Stokes equations. *Mathematics of Computation*, 22(104), 745–762.

Connell, B. S. H. & Yue, D. K. P. (2007). Flapping dynamics of a flag in a uniform stream. *Journal of Fluid Mechanics*, 581, 33.

Dellar, P. J. (2001). Bulk and shear viscosities in lattice Boltzmann equations. *Physical Review E*, 64(3), 031203.

Dowell, E. H. & Hall, K. C. (2001). Modeling of fluid–structure interaction. *Annual Review of Fluid Mechanics*, 33(1), 445–490.

Fai, T. G., Griffith, B. E., Mori, Y., & Peskin, C. S. (2013). Immersed boundary method for variable viscosity and variable density problems using fast constant-coefficient linear solvers i: Numerical method and results. *SIAM Journal on Scientific Computing*, 35(5), B1132–B1161.

Feng, Z.-G. & Michaelides, E. E. (2004). The immersed boundary-lattice Boltzmann method for solving fluid–particles interaction problems. *Journal of Computational Physics*, 195(2), 602–628.

Feng, Z.-G. & Michaelides, E. E. (2009). Robust treatment of no-slip boundary condition and velocity updating for the lattice-Boltzmann simulation of particulate flows. *Computers & Fluids*, 38(2), 370–381.

Gerbeau, J.-F. & Vidrascu, M. (2003). A quasi-Newton algorithm based on a reduced model for fluid–structure interaction problems in blood flows. *ESAIM: Mathematical Modelling and Numerical Analysis*, 37(4), 631–647.

Gingold, R. A. & Monaghan, J. J. (1977). Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 181(3), 375–389.

Ginzburg, I., Verhaeghe, F., & d'Humieres, D. (2008). Two-relaxation-time lattice Boltzmann scheme: About parametrization, velocity, pressure and mixed boundary conditions. *Communications in Computational Physics*, 3(2), 427–478.

Gombosi, T. I. & Gombosi, A. (1994). *Gaskinetic theory*. Number 9. Cambridge University Press.

Griffith, B. E., Luo, X., McQueen, D. M., & Peskin, C. S. (2009). Simulating the fluid dynamics of natural and prosthetic heart valves using the immersed boundary method. *International Journal of Applied Mechanics*, 1(01), 137–177.

Grunau, D., Chen, S., & Eggert, K. (1993). A lattice Boltzmann model for multiphase fluid flows. *Physics of Fluids A: Fluid Dynamics*, 5(10), 2557–2562.

Guo, Z. & Zhao, T. (2002). Lattice Boltzmann model for incompressible flows through porous media. *Physical Review E*, 66(3), 036304.

Guo, Z., Zheng, C., & Shi, B. (2002). Discrete lattice effects on the forcing term in the lattice Boltzmann method. *Physical Review E*, 65(4), 046308.

Gurtin, M. E., Fried, E., & Anand, L. (2010). *The mechanics and thermodynamics of continua*. Cambridge University Press.

He, X., Chen, S., & Zhang, R. (1999). A lattice Boltzmann scheme for incompressible multiphase flow and its application in simulation of Rayleigh–Taylor instability. *Journal of Computational Physics*, 152(2), 642–663.

He, X. & Luo, L.-S. (1997). A priori derivation of the lattice Boltzmann equation. *Physical Review E*, 55(6), R6333.

He, X., Shan, X., & Doolen, G. D. (1998). Discrete Boltzmann equation model for nonideal gases. *Physical Review E*, 57(1), R13.

Higuera, F. J. & Jiménez, J. (1989). Boltzmann approach to lattice gas simulations. *EPL (Europhysics Letters)*, 9(7), 663.

Hirt, C. W., Amsden, A. A., & Cook, J. (1974). An arbitrary Lagrangian–Eulerian computing method for all flow speeds. *Journal of Computational Physics*, 14(3), 227–253.

Hoover, W. G., Hoover, C., Kum, O., Castillo, V., Posch, H., & Hess, S. (2006). Smooth particle applied mechanics. *Advanced Series in Nonlinear Dynamics*, 25.

Huang, H., Krafczyk, M., & Lu, X. (2011). Forcing term in single-phase and Shan–Chen-type multiphase lattice Boltzmann models. *Physical Review E*, 84(4), 046710.

Hughes, T. J. (2012). *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation.

Judice, S. F., Coutinho, B. B. S., & Giraldi, G. A. (2010). Lattice methods for fluid animation in games. *Computers in Entertainment (CIE)*, 7(4), 1–29.

Kamrin, K. & Nave, J.-C. (2009). An Eulerian approach to the simulation of deformable solids: Application to finite-strain elasticity. *arXiv preprint arXiv:0901.3799*.

Kamrin, K., Rycroft, C. H., & Nave, J.-C. (2012). Reference map technique for finite-strain elasticity and fluid–solid interaction. *Journal of the Mechanics and Physics of Solids*, 60(11), 1952–1969.

Kandhai, D., Koponen, A., Hoekstra, A. G., Kataja, M., Timonen, J., & Sloot, P. M. (1998). Lattice-Boltzmann hydrodynamics on parallel systems. *Computer Physics Communications*, 111(1-3), 14–26.

Krüger, T., Gross, M., Raabe, D., & Varnik, F. (2013). Crossover from tumbling to tank-treading-like motion in dense simulated suspensions of red blood cells. *Soft Matter*, 9(37), 9008–9015.

Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., & Viggen, E. M. (2017). *The lattice Boltzmann method*. Springer.

Ladd, A. J. C. (1994). Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 1. Theoretical foundation. *Journal of Fluid Mechanics*, 271, 285–309.

Landau, L. D. & Lifshitz, E. M. (1987). *Fluid Mechanics, Second Edition: Volume 6 (Course of Theoretical Physics)*. Course of theoretical physics / by L. D. Landau and E. M. Lifshitz, Vol. 6. Butterworth-Heinemann.

LeVeque, R. J. (2007). *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*, volume 98. SIAM.

LeVeque, R. J. et al. (2002). *Finite volume methods for hyperbolic problems*, volume 31. Cambridge University Press.

Li, X., Vlahovska, P. M., & Karniadakis, G. E. (2013). Continuum-and particle-based modeling of shapes and dynamics of red blood cells in health and disease. *Soft Matter*, 9(1), 28–37.

Liu, C. & Walkington, N. J. (2001). An Eulerian description of fluids containing visco-elastic particles. *Archive for Rational Mechanics and Analysis*, 159(3), 229–252.

Lubliner, J. (2008). *Plasticity theory*. Courier Corporation.

Martys, N. S. & Chen, H. (1996). Simulation of multicomponent fluids in complex three-dimensional geometries by the lattice Boltzmann method. *Physical Review E*, 53(1), 743.

Mittal, R. & Iaccarino, G. (2005). Immersed boundary methods. *Annu. Rev. Fluid Mech.*, 37, 239–261.

Nie, D. & Lin, J. (2010). A LB-DF/FD method for particle suspensions. *Communications in Computational Physics*, 7(3), 544.

Osher, S. & Sethian, J. A. (1988). Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*, 79(1), 12–49.

Pan, C., Luo, L.-S., & Miller, C. T. (2006). An evaluation of lattice Boltzmann schemes for porous medium flow simulation. *Computers & Fluids*, 35(8-9), 898–909.

Persson, P.-O., Bonet, J., & Peraire, J. (2009). Discontinuous galerkin solution of the Navier–Stokes equations on deformable domains. *Computer Methods in Applied Mechanics and Engineering*, 198(17-20), 1585–1595.

Persson, P.-O. & Peraire, J. (2009). Curved mesh generation and mesh refinement using lagrangian solid mechanics. In *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition* (pp. 949).

Persson, P.-O., Peraire, J., & Bonet, J. (2007). A high order discontinuous galerkin method for fluid–structure interaction. In *18th AIAA Computational Fluid Dynamics Conference* (pp. 4327).

Peskin, C. S. (1977). Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3), 220–252.

Peskin, C. S. (2002). The immersed boundary method. *Acta Numerica*, 11, 479–517.

Plohr, B. J. & Sharp, D. H. (1988). A conservative Eulerian formulation of the equations for elastic flow. *Advances in Applied Mathematics*, 9(4), 481–499.

Qian, Y.-H., d'Humières, D., & Lallemand, P. (1992). Lattice BGK models for Navier–Stokes equation. *EPL (Europhysics Letters)*, 17(6), 479.

Rabczuk, T., Gracie, R., Song, J.-H., & Belytschko, T. (2010). Immersed particle method for fluid–structure interaction. *International Journal for Numerical Methods in Engineering*, 81(1), 48–71.

Rivet, J.-P. & Boon, J.-P. (2005). *Lattice gas hydrodynamics*, volume 11. Cambridge University Press.

Rivlin, R. S. & Saunders, D. (1951). Large elastic deformations of isotropic materials VII. experiments on the deformation of rubber. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 243(865), 251–288.

Rugonyi, S. & Bathe, K.-J. (2001). On finite element analysis of fluid flows fully coupled with structural interactions. *CMES- Computer Modeling in Engineering and Sciences*, 2(2), 195–212.

Rycroft, C. H. & Gibou, F. (2012). Simulations of a stretching bar using a plasticity model from the shear transformation zone theory. *Journal of Computational Physics*, 231(5), 2155–2179.

Rycroft, C. H., Sui, Y., & Bouchbinder, E. (2015). An Eulerian projection method for quasi-static elastoplasticity. *Journal of Computational Physics*, 300, 136–166.

Rycroft, C. H., Wu, C.-H., Yu, Y., & Kamrin, K. (2018). Reference map technique for incompressible fluid–structure interaction. *arXiv preprint arXiv:1810.03015*.

Sahimi, M. (2011). *Flow and transport in porous media and fractured rock: from classical methods to modern approaches*. John Wiley & Sons.

Sethian, J. A. (1999). *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge University Press.

Shan, X. & Chen, H. (1993). Lattice Boltzmann model for simulating flows with multiple phases and components. *Physical Review E*, 47(3), 1815.

Shan, X. & Doolen, G. (1995). Multicomponent lattice-Boltzmann model with interparticle interaction. *Journal of Statistical Physics*, 81(1-2), 379–393.

Shan, X., Yuan, X.-F., & Chen, H. (2006). Kinetic theory representation of hydrodynamics: a way beyond the Navier–Stokes equation. *Journal of Fluid Mechanics*, 550, 413–441.

Shelley, M. J. & Zhang, J. (2011). Flapping and bending bodies interacting with fluid flows. *Annual Review of Fluid Mechanics*, 43, 449–465.

Shu, C.-W. (1998). Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In *Advanced numerical approximation of nonlinear hyperbolic equations* (pp. 325–432). Springer.

Shur, M., Spalart, P., Strelets, M., & Travin, A. (1999). Detached-eddy simulation of an airfoil at high angle of attack. In *Engineering turbulence modelling and experiments 4* (pp. 669–678). Elsevier.

Shyy, W., Aono, H., Chimakurthi, S. K., Trizila, P., Kang, C.-K., Cesnik, C. E., & Liu, H. (2010). Recent progress in flapping wing aerodynamics and aeroelasticity. *Progress in Aerospace Sciences*, 46(7), 284–327.

Stomakhin, A., Schroeder, C., Chai, L., Teran, J., & Selle, A. (2013). A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, 32(4), 1–10.

Succi, S. (2001). *The lattice Boltzmann equation: for fluid dynamics and beyond*. Oxford University Press.

Succi, S., Foti, E., & Higuera, F. (1989). Three-dimensional flows in complex geometries with the lattice Boltzmann method. *EPL (Europhysics Letters)*, 10(5), 433.

Sui, Y., Chew, Y.-T., Roy, P., & Low, H.-T. (2008). A hybrid method to study flow-induced deformation of three-dimensional capsules. *Journal of Computational Physics*, 227(12), 6351–6371.

Sulsky, D., Chen, Z., & Schreyer, H. L. (1994). A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118(1-2), 179–196.

Swift, M. R., Orlandini, E., Osborn, W., & Yeomans, J. (1996). Lattice Boltzmann simulations of liquid–gas and binary fluid systems. *Physical Review E*, 54(5), 5041.

Tasora, A., Serban, R., Mazhar, H., Pazouki, A., Melanz, D., Fleischmann, J., Taylor, M., Sugiyama, H., & Negrut, D. (2015). Chrono: An open source multi-physics dynamics engine. In *International Conference on High Performance Computing in Science and Engineering* (pp. 19–49).: Springer.

Thurey, N. (2007). Physically based animation of free surface flows with the lattice boltzmann method. *Ph. D. Thesis, University of Erlangen*.

Trangenstein, J. A. & Colella, P. (1991). A higher-order godunov method for modeling finite deformation in elastic-plastic solids. *Communications on Pure and Applied Mathematics*, 44(1), 41–100.

Truesdell, C. (1955). Hypo-elasticity. *Journal of Rational Mechanics and Analysis*, 4, 83–1020.

Tryggvason, G., Bunner, B., Esmaeeli, A., Juric, D., Al-Rawahi, N., Tauber, W., Han, J., Nas, S., & Jan, Y.-J. (2001). A front-tracking method for the computations of multiphase flow. *Journal of Computational Physics*, 169(2), 708–759.

Udaykumar, H., Tran, L., Belk, D., & Vanden, K. (2003). An Eulerian method for computation of multimaterial impact with ENO shock-capturing and sharp interfaces. *Journal of Computational Physics*, 186(1), 136–177.

Valkov, B., Rycroft, C. H., & Kamrin, K. (2015). Eulerian method for multiphase interactions of soft solid bodies in fluids. *Journal of Applied Mechanics*, 82(4).

Versteeg, H. K. & Malalasekera, W. (1995). *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*. Addison Wesley Longman, Ltd.

Wang, X. S. (2008). *Fundamentals of fluid–solid interactions: analytical and computational approaches*. Elsevier.

Wang, Z., Sui, Y., Spelt, P. D., & Wang, W. (2013). Three-dimensional dynamics of oblate and prolate capsules in shear flow. *Physical Review E*, 88(5), 053021.

Watanabe, Y., Suzuki, S., Sugihara, M., & Sueoka, Y. (2002). An experimental study of paper flutter. *Journal of Fluids and Structures*, 16(4), 529–542.

Yu, J.-D., Sakai, S., & Sethian, J. A. (2003). A coupled level set projection method applied to ink jet simulation. *Interfaces and Free Boundaries*, 5(4), 459–482.

Yu, J.-D., Sakai, S., & Sethian, J. A. (2007). Two-phase viscoelastic jetting. *Journal of Computational Physics*, 220(2), 568–585.

Zhang, J., Childress, S., Libchaber, A., & Shelley, M. (2000). Flexible filaments in a flowing soap film as a model for one-dimensional flags in a two-dimensional wind. *Nature*, 408(6814), 835–839.

Zhang, J., Johnson, P. C., & Popel, A. S. (2007). An immersed boundary lattice Boltzmann approach to simulate deformable liquid capsules and its application to microscopic blood flows. *Physical Biology*, 4(4), 285.

Zhu, L., He, G., Wang, S., Miller, L., Zhang, X., You, Q., & Fang, S. (2011). An immersed boundary method based on the lattice Boltzmann approach in three dimensions, with application. *Computers & Mathematics with Applications*, 61(12), 3506–3518.

Zhu, L. & Peskin, C. S. (2002). Simulation of a flapping flexible filament in a flowing soap film by the immersed boundary method. *Journal of Computational Physics*, 179(2), 452–468.

Zienkiewicz, O. C. & Taylor, R. L. (1967). *The finite element method for solid and structural mechanics*. McGraw Hill.

Zöttl, A. & Stark, H. (2012). Nonlinear dynamics of a microswimmer in poiseuille flow. *Physical Review Letters*, 108(21), 218104.

Zou, Q. & He, X. (1997). On pressure and velocity boundary conditions for the lattice Boltzmann BGK model. *Physics of Fluids*, 9(6), 1591–1598.