

Titre: Improved Lattice Boltzmann Method for the Study of Unsteady
Title: Flows with Moving Objects

Auteur: Abdallah Mohamed Mohamed Elsherbiny
Author:

Date: 2023

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Elsherbiny, A. M. M. (2023). Improved Lattice Boltzmann Method for the Study of
Citation: Unsteady Flows with Moving Objects [Thèse de doctorat, Polytechnique Montréal].
PolyPublie. <https://publications.polymtl.ca/57072/>

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/57072/>
PolyPublie URL:

**Directeurs de
recherche:** Sébastien Leclaire
Advisors:

Programme: Génie mécanique
Program:

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

**Improved lattice Boltzmann method for the study of unsteady flows with
moving objects**

ABDALLAH MOHAMED MOHAMED ELSHERBINY

Département de génie mécanique

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*

Génie mécanique

Décembre 2023

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Cette thèse intitulée :

Improved lattice Boltzmann method for the study of unsteady flows with moving objects

présentée par **Abdallah Mohamed Mohamed ELSHERBINY**

en vue de l'obtention du diplôme de *Philosophiæ Doctor*

a été dûment acceptée par le jury d'examen constitué de :

Massimo CIMMINO, président

Sébastien LECLAIRE, membre et directeur de recherche

David VIDAL, membre

Andrea MONTESSORI, membre externe

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to those who have played a pivotal role in shaping my Ph.D. journey. Their support, guidance, and contributions have been crucial in reaching this milestone.

First and foremost, I would like to extend my deepest appreciation to my supervisor, Prof. Sébastien Leclaire. Your guidance, knowledge, and unwavering commitment to helping me learn and go forward have been incredibly precious. I am truly fortunate to have had the privilege of working under your supervision.

I owe a debt of gratitude to my esteemed team members, each of whom greatly impacts my Ph.D. study. Prof. Jean-Yves Trépanier, Nicolas Pellerin, and Sami Ammar, your high-level contributions have significantly enriched the project, and your support and presence on the team were essential. A special mention must be made to Eng. Md Sujaat Ali, who has been a colleague and friend throughout this journey. I am also thankful for the guidance and coordination provided by Prof. Mohamed Zakaria, who played a pivotal role in keeping the project on track and ensuring its success.

Last but not least, I extend my heartfelt thanks to my wife and children for their unwavering love, patience, and understanding. Your support during the long hours and sacrifices made this achievement possible. Your belief in me has been my driving force, and I am deeply grateful for your love and encouragement.

This project would not have been possible without these remarkable individuals' efforts and support. I am sincerely thankful for your impact on my life. Your contributions, in various forms, have left an indelible mark on my academic and personal journey.

With profound gratitude,

Abdallah ElSherbiny

RÉSUMÉ

Cette thèse présente une implémentation améliorée de la méthode de Boltzmann sur réseau (LBM) pour résoudre les écoulements non stationnaires autour de frontières mobiles. L'importance de l'étude de la dynamique des écoulements à proximité d'objets en mouvement ne peut être sous-estimée, compte tenu de son importance scientifique et de ses nombreuses applications pratiques dans plusieurs secteurs d'activités industriels. L'objectif principal de cette thèse est d'intégrer la LBM à une méthode conçue pour traiter les frontières mobiles. Un examen approfondi de la littérature scientifique a révélé que la méthode « overset » a déjà été intégrée à la LBM pour résoudre le problème des frontières mobiles, tout en permettant d'obtenir une exactitude de deuxième ordre. Cependant, il est apparu qu'aucune recherche antérieure n'avait mis en œuvre l'intégration d'une grille non uniforme avec l'approche Boltzmann sur réseau et « overset » (LB-overset), ce qui a mis en évidence une lacune critique en recherche.

L'utilisation d'une grille de type « quadtree », qui est une grille non uniforme, complémente efficacement la LBM en raison de ses caractéristiques d'espacement orthogonal cartésien égal. Cette structure de grille améliore l'efficacité de la simulation et réduit considérablement les ressources informatiques nécessaires en réduisant le nombre de nœuds dans le domaine de calcul. Néanmoins, l'incorporation d'une grille de type « quadtree » dans la LBM exige un traitement spécialisé lors de la propagation des fonctions de distribution. L'un de ces traitements spécialisés concerne la technique d'interpolation des fonctions de distribution post-collision afin d'effectuer la propagation, qui est généralement décrite dans la littérature comme nécessitant une interpolation spatiale, telle que l'interpolation bicubique, ce qui ajoute de la complexité et augmente les exigences en matière de calcul.

Par conséquent, la lacune identifiée dans la recherche est l'absence d'études mettant en œuvre la méthode LB-overset avec raffinement de grille de type « quadtree » basée sur la technique d'interpolation des fonctions de distribution post-collision. Par conséquent, cette lacune dans la recherche définit directement l'objectif principal de cette thèse : vérifier et valider la méthode LB-overset lorsqu'elle est couplée avec une grille « quadtree » et l'approche d'interpolation des fonctions de distribution post-collision.

L'objectif principal est atteint grâce à la réalisation de trois étapes clés. L'étape initiale consiste à mettre en œuvre la LBM standard avec diverses conditions aux limites, y compris le

traitement des frontières courbes. Cette phase couvre également les aspects fondamentaux de la conception des structures de données, de la description des conditions limites et de la détection des nœuds frontières, solides et fluides.

Ensuite, la deuxième étape consiste à intégrer et à examiner la LBM avec la méthode « overset », qui sert d'étape intermédiaire avant d'incorporer le maillage « quadtree ». La méthode « overset » sépare le domaine en une grille stationnaire dans le repère inertiel (grille fixe) et une grille associée à des frontières mobiles (grille mobile). Ces grilles communiquent par le biais d'interpolations aux frontières, et des investigations approfondies sur l'interpolation aux frontières sont menées, l'interpolation bicubique donnant les résultats les plus précis.

Dans l'étape finale, le maillage quadtree est intégré dans la méthode LB-overset, avec l'ajout de l'approche d'interpolation des fonctions de distribution post-collision afin d'effectuer la propagation. Cette intégration nécessite la transformation de la grille mobile en un maillage quadtree, tandis que la grille fixe reste une grille uniforme avec le déplacement spatial maximal de la grille mobile. En outre, cette thèse apporte une amélioration à l'approche d'interpolation des fonctions de distribution post-collision en exploitant les caractéristiques lagrangiennes de la LBM. Cette thèse introduit des méthodes d'interpolation basées sur l'interpolation de liens lagrangiens à 1 dimension, formant ainsi pour la propagation ce que l'on appelle le schéma d'interpolation « spatial Lagrangian-link » des fonctions de distribution post-collision. Ce schéma applique initialement l'interpolation spatiale en tant qu'étape prédictive. Ensuite, cette valeur interpolée par le prédicteur spatial est utilisée pour l'interpolation du lien lagrangien en tant que correcteur. Ces innovations réduisent considérablement les ressources informatiques nécessaires et améliorent l'efficacité de la simulation. En outre, une technique de séparation des nœuds est développée pour accroître l'efficacité du schéma d'interpolation proposé.

Pour finir, la méthode intégrée fait l'objet d'une de vérifications et de vérifications code à code approfondies à l'aide de divers cas tests, tout en conservant une exactitude de deuxième ordre sans dégradation. Par ailleurs, les paramètres clés tels que le coefficient de portance, le coefficient de traînée et le nombre de Strouhal correspondent étroitement aux données publiées précédemment. En réalisant cette troisième et dernière étape, l'objectif principal de cette thèse est atteint.

ABSTRACT

This thesis presents an enhanced implementation of the Lattice Boltzmann method (LBM) for solving unsteady flow around moving boundaries. The significance of addressing flow dynamics in the proximity of moving objects cannot be understated, given its profound scientific significance and numerous practical applications across industries. The primary objective of this thesis is to integrate the LBM with a method designed for handling moving boundaries. A comprehensive literature review revealed that the overset method was previously integrated with LBM to address the challenge of moving boundaries, which retrieves second-order accuracy. However, it became apparent that no previous research had implemented the integration of a non-uniform grid with the Lattice Boltzmann overset (LB-overset) approach, thus highlighting a critical research gap.

The utilization of a quadtree grid, which is a non-uniform grid, complements the LBM effectively due to its orthogonal cartesian equal spacing characteristics. This grid structure enhances simulation efficiency and significantly reduces the computational resources required by reducing the number of nodes within the computational domain. Nevertheless, incorporating a quadtree grid into LBM demands specialized handling of particle distribution functions' streaming. One of these specialized treatments involves the interpolated post-collision PDF streaming technique, which is commonly described in the literature as necessitating spatial interpolation, such as bicubic interpolation, thereby adding complexity and heightened computational demands.

Hence, the identified research gap is the absence of studies implementing the LB-overset method with quadtree grid refinement based on the interpolated PDF streaming approach. Consequently, this research gap directly defines this thesis's main objective: to verify and code-to-code verify the integrated Lattice Boltzmann method with the moving overset approach, coupled with quadtree grid refinement based on the interpolated PDF streaming scheme.

The main objective is achieved through the execution of three key steps. The initial step involves the implementation of the standard LBM encompassing various boundary conditions, including the treatment of curved boundaries. This phase also covers fundamental aspects of data structure design, description of boundary conditions, and detecting boundary, solid, and fluid nodes.

Subsequently, the second step involves integrating and examining LBM with the overset method, which serves as an intermediary stage before incorporating the quadtree mesh. The overset method segregates the domain into a stationary grid in the inertial frame (fixed grid) and a grid associated with moving boundaries (moving grid). These grids communicate through border interpolations, and extensive investigation into border interpolation is conducted, with bicubic interpolation yielding the most accurate results.

In the final step, the quadtree mesh is integrated into the LB-overset method, with the addition of the interpolated post-collision PDF streaming approach. This integration necessitates the transformation of the moving grid into a quadtree mesh, while the fixed grid is still a uniform grid with the maximum moving grid spatial displacement. Furthermore, this thesis introduces an enhancement to the interpolated post-collision PDF streaming approach by exploiting the Lagrangian characteristics of the LBM nature. It introduces interpolation methods based on the 1-D Lagrangian link interpolation, thus forming what is referred to as the spatial Lagrangian-link interpolation post-collision PDF streaming scheme, which initially applies spatial interpolation as a predictor step. Subsequently, this spatial predictor-interpolated value is utilized for a Lagrangian-link corrector interpolation. These innovations significantly reduce the computational resources required and enhance simulation efficiency. Additionally, a node-splitting technique is developed to further increase the efficiency of the proposed interpolation scheme.

Furthermore, the integrated method undergoes thorough verification and code-to-code verification through diverse test cases, maintaining a second-order accuracy without degradation. Also, key parameters such as the lift coefficient, drag coefficient, and Strouhal number align closely with previously published data. By completing this third and final step, the thesis's primary objective is accomplished.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	III
RÉSUMÉ.....	IV
ABSTRACT	VI
TABLE OF CONTENTS	VIII
LIST OF TABLES	XII
LIST OF FIGURES	XIII
LIST OF ABBREVIATIONS	XVII
CHAPTER 1 INTRODUCTION	1
1.1 Computational Fluid Dynamics.....	2
1.2 Boundary treatment based on grid modification	4
1.3 Moving boundary treatment	5
1.4 Summary.....	6
CHAPTER 2 LITERATURE REVIEW	8
2.1 Lattice Boltzmann Method	8
2.2 Boundary treatment based on grid modification	11
2.3 Moving boundary treatment	14
CHAPTER 3 RESEARCH GAP, OBJECTIVES, AND OUTLINE	21
3.1 Research Gap.....	21
3.2 Objectives	22
3.3 Outlines.....	23
CHAPTER 4 TECHNICAL CHAPTER	25
4.1 Standard LBM	25
4.2 Boundary conditions.....	27

4.2.1	Zou and He velocity inlet/ pressure outlet	27
4.2.2	Straight velocity boundary conditions (Inamuro)	29
4.2.3	Halfway bounce-back.....	30
4.2.4	Curved boundary treatment (Guo's extrapolation method)	31
4.3	Mesh generation	33
4.3.1	Data structure	34
4.3.2	Mesh generation algorithm.....	36
4.3.3	Detect solid nodes	36
4.3.4	Detect boundary Cells and nodes	37
4.3.5	Displacement calculation	40
4.3.6	Mesh generation algorithm Procedures	42
4.4	LBM with extrapolation technique implementation.....	43
4.5	Results and discussion.....	45
4.5.1	Poiseuille flow	45
4.5.2	Taylor-Couette flow	48
4.5.3	Uniform flow around a circular cylinder.....	49
4.5.4	Uniform flow around rotating circular cylinder	51
CHAPTER 5 ARTICLE 1: IMPACT OF INTERPOLATION METHOD SELECTION ON THE ACCURACY OF THE LATTICE BOLTZMANN-OVERSET METHOD.....		55
5.1	Abstract.....	55
5.2	Introduction	55
5.3	The Lattice Boltzmann method using the recursive regularized BGK model.....	57
5.4	The lattice Boltzmann overset method with recursive regularized BGK model	60
5.4.1	Fixed-to-moving treatment.....	62
5.4.2	Moving-to-fixed treatment	63

5.5	Computational Procedures.....	63
5.6	Interpolation algorithm	64
5.6.1	Stencil detection algorithm.....	65
5.6.2	Interpolators	65
5.7	Result and discussion.....	70
5.7.1	Uniform flow.....	71
5.7.2	Poiseuille flow	75
5.7.3	Taylor-Couette flow	79
5.7.4	High Reynolds advection vortex in uniform flow.....	82
5.7.5	Uniform flow around a cylinder.....	84
5.7.5.1	Uniform flow a cylinder with a non-rotating grid	85
5.7.5.2	Uniform flow around a rotating cylinder.....	86
5.8	Conclusion.....	88
5.9	APPENDIX A.....	90
5.9.1	Hermite polynomials and higher-order non-equilibrium distributions	90
5.9.2	Interpolators verification	90
CHAPTER 6 ARTICLE 2: INTEGRATION OF LATTICE BOLTZMANN-OVERSET METHOD WITH NON-CONFORMING QUADTREE MESH BASED ON THE COMBINATION OF SPATIAL AND LAGRANGIAN-LINK INTERPOLATED STREAMING TECHNIQUE		93
6.1	Abstract.....	93
6.2	Introduction	94
6.3	The recursive regularized LB-overset method	99
6.3.1	Recursive regularized LBM	99
6.3.2	LB-overset method.....	101

6.4	Non-conforming quadtree mesh based on interpolated PDF streaming technique	104
6.4.1	Quadtree mesh and streaming nodes' splitting	104
6.4.2	Lagrangian-link interpolated PDF streaming technique	106
6.4.3	Spatial Lagrangian-link interpolated PDF streaming technique	107
6.5	The Result and Discussion	109
6.5.1	Uniform flow	109
6.5.2	Poiseuille flow	112
6.5.3	Taylor-Couette flow	114
6.5.4	Flow around a rotating cylinder	116
6.6	Conclusion	122
CHAPTER 7 GENERAL DISCUSSION		124
CHAPTER 8 CONCLUSION AND RECOMMENDATIONS		128
8.1	Conclusion	128
8.2	Limitations	129
8.3	Future research and recommendations	129
REFERENCES		130

LIST OF TABLES

Table 4.1 Poiseuille flow Test cases for different boundary conditions.	45
Table 4.2 Poiseuille flow $e2u$ and order of accuracy for different boundary conditions.	46
Table 4.3 Poiseuille flow $e2u$ and order of accuracy for case (a) with $\tau = 0.7$	47
Table 4.4 The lift and drag coefficients and Strouhal number for $Re=100$	51
Table 4.5 The lift and drag coefficients for rotation ratio = 0.5 and $Re=20$	53
Table 4.6 The lift and drag coefficients and Strouhal number for rotation ratio = 1.0 and $Re=100$	53
Table 5.1. The $L1$ norm error and pressure noise for the introduced interpolators after grid rotation at tf	73
Table 5.2. The velocity discrepancy and pressure noise for the introduced interpolators.	78
Table 5.3. The velocity error for the introduced interpolators.	81
Table 5.4. The density discrepancy for the introduced interpolators at $FTT=5$	84
Table 5.5. The lift and drag coefficients and Strouhal number for $Re=100$	86
Table 5.6. The lift and drag coefficients and Strouhal number for rotation ratio = 0.1 and $Re=60$	88
Table 5.7. The lift and drag coefficients and Strouhal number for rotation ratio = 0.5 and $Re=20$	88
Table 5.8. Interpolators verification table. The zero values in the tables represent the machine's accuracy.	92
Table 6.1. The $L1$ norm error and pressure noise after grid rotation at tf	112
Table 6.2. The Poiseuille flow $L1u$, $\partial P \partial t$, $L2u$, and order of accuracy at tf	114
Table 6.3. The lift and drag coefficients and Strouhal number.	118
Table 7.1. The computational cost for the LB-overset and quadtree LB-overset methods.	126

LIST OF FIGURES

Figure 1.1 The unsteady aerodynamic problem, showing the main solving branches and their corresponding methods.	3
Figure 1.2 LBM boundary treatment hierarchy.	5
Figure 2.1 Collision step.	9
Figure 2.2 The streaming step and the PDFs	10
Figure 2.3 continues the boundary treatment hierarchy shown in Figure 1.2.	14
Figure 2.4 Moving boundary treatments hierarchy.	14
Figure 2.5 Illustration of the Eulerian fluid grid and the solid arbitrary lagrangian nodes of the IB-LBM.	15
Figure 2.6 Multi-block movement in IB-LBM with the solid body motion presented for different times.	16
Figure 2.7. The difference in the grid communication between the sliding mesh boundary and the moving overset-LBM method.	19
Figure 3.1. The literature survey's overall picture and the research gap.	22
Figure 4.1. The Flow chart of the standard LBM.	27
Figure 4.2. The unknown post-streaming particle distribution function in the inlet and outlet boundaries.	28
Figure 4.3. The unknown post-streaming particle distribution function in the upper boundary... ..	29
Figure 4.4. The timing difference between the full-way and half-way bounceback schemes.	30
Figure 4.5. Schematic showing the location of the wall at the half of spatial displacement on a node-base computational domain for the halfway bounce-back boundary condition.	31
Figure 4.6. Schematic showing extrapolation technique different locations and their corresponding variables.	32

Figure 4.7. Schematic showing the indices and directions for the Nodes array and Surrounding array.....	34
Figure 4.8. Schematic showing winding number algorithm.	36
Figure 4.9. Schematic showing the twelve possible combinations of the solid and fluid nodes inside a boundary cell.	39
Figure 4.10. A visualization of solid, boundary, and fluid nodes for NACA 2412 airfoil.	40
Figure 4.11. Schematic showing the intersection points required for the displacement function calculations.....	41
Figure 4.12. Schematic showing the intersection points required for the displacement function calculations.....	41
Figure 4.13. Intersection point schematic using two lines intersection method.	42
Figure 4.14. The flow chart of the mesh generation algorithm.....	43
Figure 4.15. The flow chart of the LBM with the extrapolation technique.	44
Figure 4.16. Numerical and analytical velocity profile and field for grid size 128×64 with halfway bounce-back. Other case has similar profiles.....	46
Figure 4.17. Taylor-Couette flow numerical and analytical velocity profile and field for grid size 256×256	49
Figure 4.18. Taylor-Couette order of accuracy.	49
Figure 4.19. Velocity field for flow around rotating cylinder at $Re=100$	50
Figure 4.20. Velocity field for flow around rotating cylinder at $Re=20$, $a=0.5$	52
Figure 4.21. Velocity field for flow around rotating cylinder at $Re=100$ and $a=1.0$	52
Figure 5.1. The LB-overset method with the illustration of the fixed and moving grids and the communicating borders.	61
Figure 5.2. Schematic for moving-to-fixed borders stencil detection algorithm, showing the four nodes stencil, the sixteen nodes stencil, the target node located on the border, and winding number algorithm stencil.....	65

Figure 5.3. Schematic showing the indexing defined for all interpolators.	66
Figure 5.4. Schematic showing the four interpolations in the x-direction implemented at the target node projection and the interpolation in the y-direction implementing the total interpolation on the target node.	69
Figure 5.5. The uniform flow velocity field is shown on the overlapped grids implemented at tf for the bicubic interpolator.	72
Figure 5.6. The pressure noise $\partial P / \partial ttf$ showed on the fixed grid after rotation..	74
Figure 5.7. An example of the velocity field for the bicubic interpolator at tf is presented in the overlapped grids.	76
Figure 5.8. The velocity profile comparison of the different interpolators before and after the moving grid at tf	77
Figure 5.9. Moving grid velocity field at tf for bicubic interpolator (a) and nearest neighbor interpolator (b).	78
Figure 5.10. An example of the velocity field for the bilinear interpolator is presented in the overlapped grids.	80
Figure 5.11. The velocity profile for the bilinear interpolator at tf	81
Figure 5.12. The initial condition of the density field is shown in the two grids.	83
Figure 5.13. The vortex density profile for different interpolators and the reference test case (without overset) at FTT=5.	83
Figure 5.14. The velocity field is shown in the overlapped grid for the bicubic interpolator at tf	87
Figure 5.15. Schematic showing the stencil and the sample points for the interpolators' verification process.	91
Figure 6.1 Schematic shows the quadtree mesh splitting.	104
Figure 6.2 An example of a circular boundary shows the grid nodes' classification as solid, fluid, and boundary nodes.	105
Figure 6.3 Schematic shows the three classifications of the streaming nodes.	106

Figure 6.4 The interpolated post-collision PDFs.	107
Figure 6.5 Schematics showing the spatial Lagrangian-link interpolation.	108
Figure 6.6 The mesh refinement based on sinusoidal wave control points.....	110
Figure 6.7 Velocity profile after the moving grid at 75% of the channel length for the test case of 128×64 at tf	113
Figure 6.8 Velocity field of the overlapped grids for the test case of 128×64 at tf	113
Figure 6.9 Taylor-Couette flow velocity field of the overlapped grids at tf	116
Figure 6.10 Different layers of control points for flow around a cylinder. The presented figure is coarser than the one used in the simulation for visualization purposes.	117
Figure 6.11 An example of the moving quadtree grid streaming nodes splitting.	117
Figure 6.12 Comparison of the velocity profile using the Bilinear PDF streaming and the spatial Lagrangian-link PDF streaming for $Re = 100$ and $a = 1$	120
Figure 6.13 Comparison of the velocity profile of the moving grid regarding the steady and the unsteady solution depending on the rotational ratio.....	121
Figure 6.14 Comparison of the velocity profile using the LB-overset method with and without the quadtree spatial Lagrangian-link PDF streaming for $Re = 20$ and $a = 0.5$	122

LIST OF ABBREVIATIONS

ALE	Arbitrary Lagrangian-Eulerian
BGK	Bhatnagar-Gross-Krook
CFD	Computational Fluid Dynamics
FDM	Finite Difference Method
FEM	Finite Element Method
FSI	Fluid-Structure Interaction
FVM	Finite Volume Method
IBM	Immersed Boundary Method
LBE	Lattice Boltzmann Equation
LBM	Lattice Boltzmann Method
MCVLBM	Mass-Conserved Volumetric LBM
MRT	Multiple Relaxation Time
PDF	Particle Distribution Function
NSCG	Non-uniform Staggered Cartesian Grid
RRLBM	Recursive Regularized Lattice Boltzmann Method
SRT	Single Relaxation Time
SPH	Smoothed Particle Hydrodynamics

CHAPTER 1 INTRODUCTION

Many failures in developing new aircraft designs can often be attributed to inaccuracies in aerodynamic calculations. One potential contributing factor is the omission of considerations for flow unsteadiness, a phenomenon prevalent in critical applications like flap movement, the oscillation of control surfaces, the separation of airplane stores, and the operation of wind turbines and rotary wings. First, regarding the store separation, the incorrect aerodynamic calculations of the stores' movements could lead to crashing the stores with each other or even colliding with the airplane body itself [1]. Second, the wind turbine's efficiency could be increased by studying the blades' unsteady response due to the cyclic variation of incidence [2] and different wind conditions applied to vertical/horizontal wind turbines [3, 4]. Third, flaps and control surfaces are crucial for an airplane or unmanned aerial vehicle. Their oscillating movements cause an unsteady flow around the wings. So, this unsteadiness must be studied to avoid incorrect aerodynamic data such as stability and controllability derivatives. Last but not least, flaps in rotor wings play an essential role in the smart-structure concept by varying the airfoil chord at different blade positions during the rotation, maximizing the lift generated from the rotor, which also could be applied in optimizing the wind turbine designs [5].

There are three main branches of studying the unsteady flows of moving objects. These branches are analytical/theoretical, experimental, and computational. The theoretical unsteady flows methods have low fidelity but are very simple and produce instantaneous results such as the Theodorsen model [6-8], Wagner step response [6, 9], State-space finite model, and the unsteady vortex lattice method [10]. For example, the Theodorsen model mainly depends on the potential flow, while the Wagner step response applied a linear step input on a flat plate with a state-space finite model, an approximate expression of the Wagner step response. Moreover, most of these methods obtain the circulation and lift coefficient around airfoils with the effective angle of attack changing but do not study the drag coefficients or the overall flow field. Furthermore, most of these methods are applied only to two-dimensional flows, which cannot be extended to three-dimensional flows.

On the other hand, the experimental unsteady flows branch evaluates the most accurate results. However, it requires special tools and equipment, such as wind tunnels and large workgroups. There are various special wind tunnels; each depends on the studied applications. For

instance, the moving objects studies must be applied on a wind tunnel equipped with a captive trajectory system to calculate, translate, and rotate the model to the new position [11]. Moreover, most existing wind tunnels are low subsonic, limiting the study regimes. Furthermore, the low availability of wind tunnels limits the researchers' usage. Finally, each experiment is expensive, mainly when applied to more special cases, such as supersonic, transonic, and hypersonic tests.

The third branch is computational fluid dynamics (CFD); in contrast with the experimental work, it does not require special equipment except a good processing machine. Moreover, it is frequently more accurate and complex than the theoretical solutions but costs much more computational time than the latter. Also, in many cases, its results are matched to the wind tunnel results [12, 13]. Furthermore, CFD can be used for different exceptional cases with no more special equipment or cost. In fact, the cost of the computational simulation goes down with time due to the rapid advancement in computer production, while the experimental costs increase with time as the cost increases in materials and labor salary. One advantage of the experimental work over the numerical one is that it may be more accurate and realistic because the numerical method may suffer from many error sources, such as modeling assumptions, discretization errors, and numerical errors. The discretization and numerical error could be reduced by selecting the proper scheme and mesh size. Furthermore, the mesh type and solving scheme selection are crucial, as selecting the best method and mesh size depending on the required application may lead to excellent results and is less time-consuming than inappropriate mesh type selection.

1.1 Computational Fluid Dynamics

Computational fluid dynamics is an essential discipline in fluid mechanics that transforms the fluid dynamics physical problem into discrete numerical models. This involves formulating governing partial differential equations, such as the Navier-Stokes equations describing fluid flow behavior. By discretizing these equations, CFD transforms complex fluid dynamics problems into a system of algebraic equations that can be solved computationally. CFD problems and their solving schemes can be categorized depending on the problem scale length, divided into macroscopic, mesoscopic, and microscopic scales, as shown in Figure 1.1.

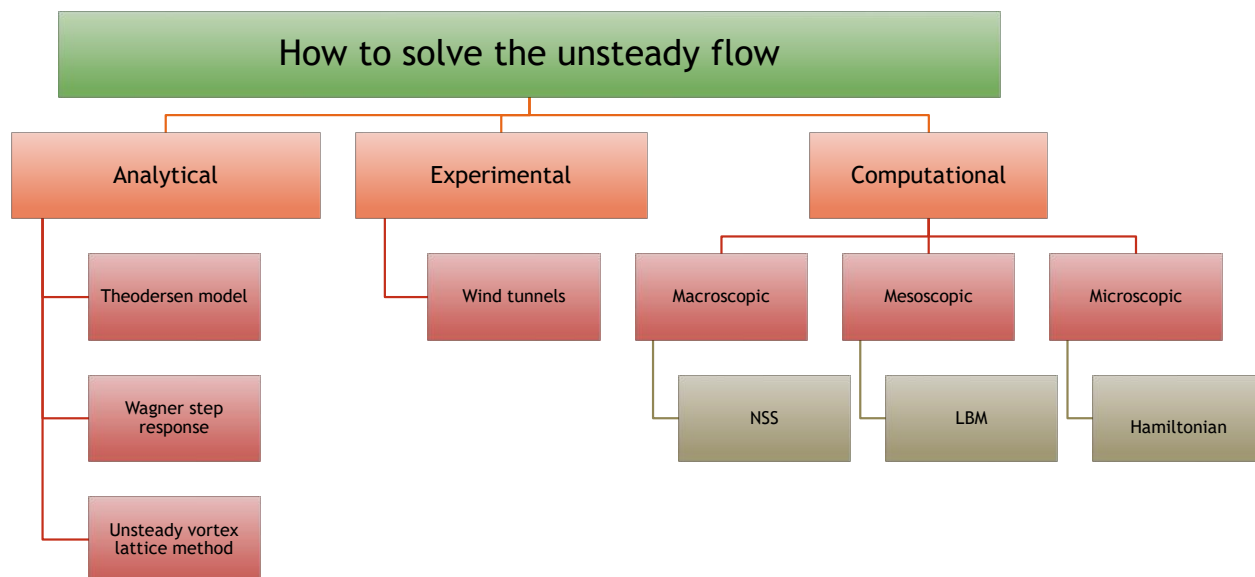


Figure 1.1 The unsteady flows problem, showing the main solving branches and their corresponding methods.

First, the macroscopic scale depends on the continuum mechanics that aim to solve Navier-Stokes equations in classical methods such as finite difference (FDM), finite volume (FVM), and finite element (FEM) or depending on the lagrangian approach such as the smoothed particles hydrodynamics method (SPH). The Navier-Stokes equation is described as a set of partial differential equations where the conservation laws of mass, momentum, and energy are the governing equations that describe the fluid flow system besides the state principle of equilibrium thermodynamics equation.

The FDM and FVM are frequently used in CFD, but the FEM is primarily used in structural mechanics [14]. The FDM is mainly used with a uniform grid, as in the case of the non-uniform grid, it mostly requires a transformation of the physical space into a uniform grid computational space, and the governing equations are discretized with a selected finite difference approximation, then they are solved numerically [15]. Contrarily, FVM does not require any transformation, as the governing equations are solved in the integral form [16].

The SPH method is considered an emerging unmaturred Navier-Stokes Lagrangian numerical technique that still needs more investigation in some essential characteristics such as convergence, numerical stability, and boundary conditions [17]; thus, the SPH research and engineering international community (SPHERIC) exists to solve these issues and make it a mature method [18]. However, it is very promising as it is a mesh-free method, making it preferable to

handle large fluid deformations such as waves. It is also adaptable for parallelization and GPU processing, and the particle movements depend on a selected Dirac function affecting the surrounding particles.

Secondly, the mesoscopic scale presented in the lattice Boltzmann method (LBM) depends on the statistical mechanics and aims to solve a simplified Boltzmann equation. The simplified lattice Boltzmann equation mainly evaluates the microscopic variables presented in the particle distribution functions (PDF) defined in the discrete lattice space, which describes the distribution of the fluid particles as an ensemble of populations (i). Then, it retrieves the macroscopic variables presented in the fluid properties, such as velocity and density, based on the kinetic theory by integrating the particle distribution function moments over the lattice space [19].

Finally, the microscopic scale depends on the Hamiltonian mechanics [20] and aims to describe the fluid flow using the Hamilton equations. It is commonly applied to multibody dynamics and has some drawbacks regarding complexity and computational efficiency [21].

The classical methods like FDM, FVM, and FEM may be challenging in parallelization as it is usually performed through distributing the computational workload across multiple processors, such as the domain decomposition technique [22, 23], and they ultimately depend on the meshing quality [24]. On the other hand, the standard LBM is an emerging method with straightforward procedures that are easy to parallelize and use uniform orthogonal mesh. Nevertheless, it requires more simulation resources than the classical methods, but parallelization can overcome this issue. Furthermore, LBM has considerable interest from the scientific community due to some advantages in kinetic behavior, its importance in various industrial and academic applications due to adaptation to various physics, and retrieving second-order accuracy in both space and time [16, 25-28]. Moreover, it is suitable for solving other new physics phenomena and has a wide range of multiphase and multicomponent applications [29]. Also, The LBM is considered a mature method as the Navier-Stokes equation could be recovered from the lattice Boltzmann equation [28].

1.2 Boundary treatment based on grid modification

One of the LBM challenges is the treatment of boundary conditions. The past decade has seen the rapid development of boundary treatments in many approaches, such as the bounceback approach, non-equilibrium extrapolation approach [30], and interpolation with the momentum exchange approach [31, 32]. However, other studies present boundary treatments based on grid

modification, such as mesh body fitting and orthogonal grid refinement. The body fitting methods include the body-conforming curvilinear mesh method [22, 23] and LBM unstructured meshing methods [33-38]. Moreover, the LBM unstructured meshing methods are also considered non-orthogonal grid refinement methods. The orthogonal grid refinement methods can be classified into uniform [25, 39, 40] and non-uniform orthogonal grids [41-44], as shown in Figure 1.2.

The body-fitting methods eliminate any interpolation or extrapolation boundary treatment methods by conforming to the boundary. It gives accurate results as the presence of the non-uniform mesh that conforms to the non-regular boundaries. However, it adds more complex data structures, complex grid generation, and increased computational resources due to additional calculations, contradicting LBM's simplicity advantages on regular grids, especially for unstructured grid methods.

In comparison, the orthogonal grid refinement aims to refine the meshes around the boundaries based on orthogonal cells, decreasing the required computational resources by decreasing the number of nodes. However, it requires some interpolation or extrapolation methods to add the effect of the boundary, as these methods do not conform to the boundary.

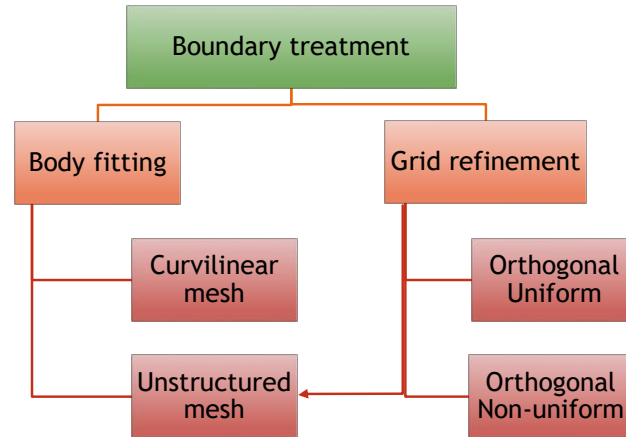


Figure 1.2 LBM boundary treatment hierarchy.

1.3 Moving boundary treatment

The moving boundary treatments have long been a question of great interest with various computational fluid dynamics solvers. This study investigates the unsteady flow problem of moving objects; thus, studying the moving boundary treatments is essential. These treatments could be classified into four main methods applied with LBM: immersed boundary LBM (IB-LBM) [45-

51], mass-conserved volumetric LBM (MCVLBM) [52], arbitrary Lagrangian-Eulerian LBM (ALE-LBM) [53, 54], and LB-overset method [55, 56].

The IB-LBM represents the fluid flow within an Eulerian Cartesian grid, with boundaries immersed as arbitrary Lagrangian nodes. Both types of nodes are integrated within a unified coordinate system. This approach is characterized by its simplicity and its first-order spatial resolution accuracy. It is widely used in fluid-structure interaction problems, such as the deformation of nanofluid droplets [50]. The MCVLBM is a volume-based rather than a node-based lattice, as fluid particles are evenly distributed within every lattice cell instead of being located exclusively at the grid nodes as in the standard LBM. The boundary movement is performed using arbitrary nodes and treated using a PDF boundary-induced migration process. It is implemented on FSI problems such as the simulation of blood flow in Aoerta [52].

The ALE-LBM method is an approach that deals with fluid flow in two different grids: one that remains stationary as a reference frame and another that moves to add the effect of boundaries' motion. This method handles fluid data in nodes of both grids, all aligned with a common reference coordinate system, the stationary frame. The data is transformed between the two grids by interpolating macroscopic variables over the entire moving grid.

Similarly, the LB-overset method has two grids: fixed and moving. However, each grid has its own coordinate system. The boundary is attached to the moving grid; consequently, any motion of the boundary leads to the motion of the moving grid. Moreover, the data is transformed between the two grids using interpolation on two boundaries. Furthermore, some modifications are applied to the interpolated data as the two flows are solved in two different coordinate systems. Both the ALE-LBM and LB-overset methods are applicable to rigid boundary motion problems.

1.4 Summary

The goal of this study is to combine the LBM with a moving boundary treatment method and a grid refinement method and ensure it is correctly integrated through verification and code-to-code verification. The verification and code-to-code verification will be implemented through different 2D test cases as it is easier to verify and validate a new combined method in 2D than in 3D.

This chapter briefly introduced the unsteady flows problem with a moving boundary and its applications. The applications presented early in this chapter share the same physical problem: rigid body motion in a fluid medium. Therefore, this study focuses on solving the 2D unsteady fluid flows affected by moving boundaries rather than studying a particular application. Moreover, adding a grid refinement approach will reduce the required computational resources by decreasing the number of nodes. Thus, this chapter briefly introduced the different methods of the CFD solvers, the moving boundary treatment methods, and the grid refinement approaches. The selection of the moving boundary treatment method and the grid refinement method are based on the literature survey, which is discussed in the following chapter.

CHAPTER 2 LITERATURE REVIEW

This chapter illustrates the different methods of solving the unsteady flow with moving boundary using LBM. First, the LBM core is introduced, providing the reader with the information needed for the literature review. Then, the previous studies of the LBM boundary treatments are discussed as it is one of the most crucial issues in LBM. Finally, the previous studies of the moving boundary treatments are illustrated as our main problem is to solve the flow around moving objects.

2.1 Lattice Boltzmann Method

The lattice Boltzmann method is an emerging matured CFD solver based on the principles of kinetic theory. It involves discretizing the lattice Boltzmann equation (LBE) in time and space, utilizing a discretized velocity space denoted as (ξ_i) . This discretization is achieved through Gauss-Hermite quadrature, enabling the evaluation of the particle distribution function $f_i(\vec{x}, t)$ at a specific location, time, and lattice direction. The LBM's core operation revolves around two primary steps: collision and streaming.

The collision step is a relaxation process of the particle distribution function $f_i(\vec{x}, t)$ towards the equilibrium PDF $f_i^{eq}(\vec{x}, t)$ after a specific time called relaxation time (τ). In other words, the collision step describes the collision of the PDF $f_i(\vec{x}, t)$ in a specific location, then redefining them with new values based on their equilibrium PDF $f_i^{eq}(\vec{x}, t)$, as shown in Figure 2.1.

The equilibrium PDF is typically defined using the Maxwell-Boltzmann distribution, and its formula change depends on the lattice model. For instance, the D2Q9 model is applied for two dimensions flows with population discretized into nine directions ($i = 0: 8$), and the D3Q19 model for three dimensions flows with population directions ($i = 0: 18$). Moreover, the equilibrium PDF is calculated using the weighting function ω_i and the local microscopic velocity (ξ_i) associated with the lattice direction i , and the macroscopic variables presented in the flow density (ρ) and velocity ($\vec{u}$) as $f_i^{eq}(\vec{x}, t) = w_i \rho \left(1 + \frac{\xi_i \cdot \vec{u}}{c_s^2} + \frac{(\xi_i \cdot \vec{u})^2}{2 c_s^4} - \frac{(\xi_i \cdot \vec{u})^2}{2 c_s^2} \right)$.

The most common collision model is the linearized Bhatnagar–Gross–Krook BGK model, which calculates the post-collision PDF $f_i^*(\vec{x}, t)$ using the equilibrium PDF and the relaxation time as $f_i^*(\vec{x}, t) = f_i(\vec{x}, t) - \frac{f_i(\vec{x}, t) - f_i^{eq}(\vec{x}, t)}{\tau}$. Moreover, the BGK collision model recovers the continuity

and momentum equations using Chapman–Enskog analysis as a function of the PDF (f_i) and microscopic velocity (ξ_i) [57] as $\rho = \sum_i f_i$, $\rho u = \sum_i f_i \xi_i$. This leads to retrieving the macroscopic variables, the flow density (ρ) and velocity ($\vec{u}$). Moreover, the pressure (P) is calculated using the LBM equation of state for ideal isothermal flow $P = \rho c_s^2$, where (c_s) is the lattice celerity. This lattice celerity is the speed at which the information propagates in the lattice. $c_s = \Delta x / (\sqrt{3} \Delta t)$, also called the speed of sound in the lattice, where Δx and Δt are the space displacement and time step, respectively.

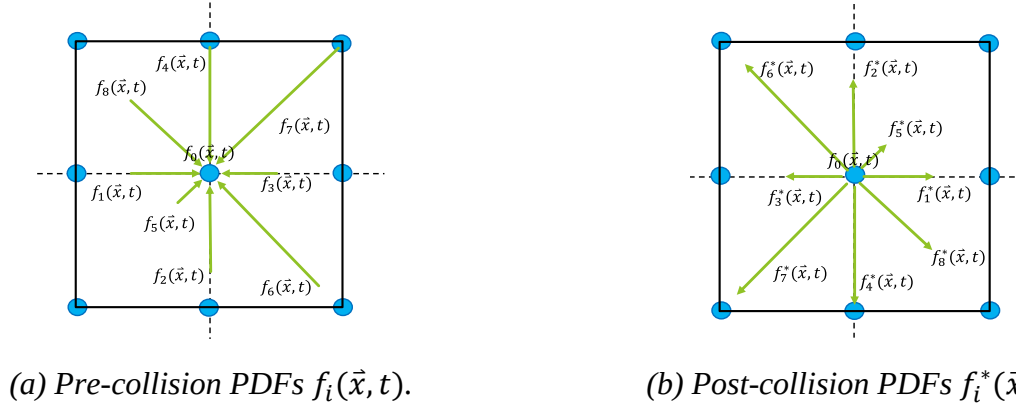
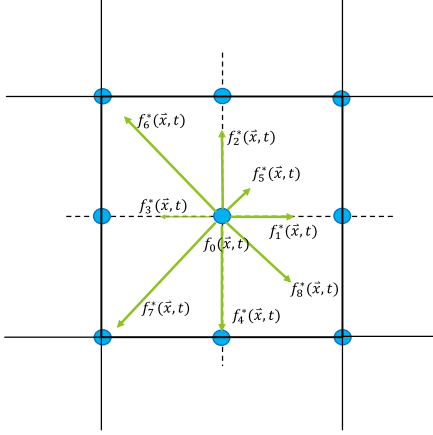


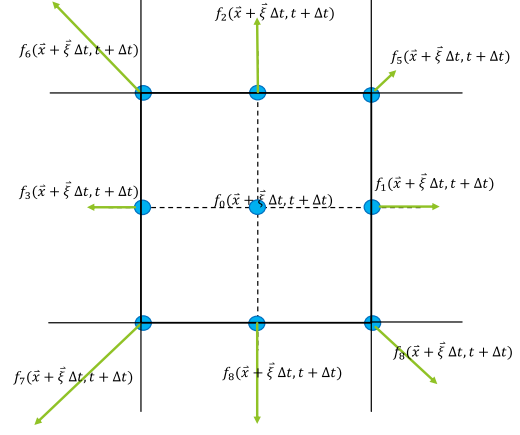
Figure 2.1 Collision step. The blue solid circles are the grid nodes, and the green arrows are the particle distribution functions.

The streaming step represents the propagation of the post-collision PDFs to the surrounding nodes, each in its direction (i), which obtain the post-streaming PDFs on these nodes, as shown in Figure 2.2. In the following time step, the previous post-streaming PDFs are the current pre-collision ones.

The most commonly used system of units in LBM is the so-called lattice system of units. In this system, the physical parameters are transformed from the physical domain to the lattice domain to ensure that the time step and node spacing equal one ($\Delta t = \Delta x = 1$). Thus, this system decreases the effort in coding the LBM core. However, similarity calculations must be performed to ensure the exact value of the non-dimensional parameters. In our case, it is the Reynold number that is usually of importance $Re = u L / \nu$. For instance, the freestream velocity (u) will be changed based on the change of the dynamic viscosity (ν) for a specific value of the relaxation time $\tau = 3\nu \Delta t / \Delta x^2 + 0.5$, to maintain the exact value of the Reynolds number in both the physical and the lattice system of units.



(a) Pre-streaming (post-collision) PDFs.



(b) Post-streaming (pre-collision) PDFs.

Figure 2.2 The streaming step and the PDFs

The standard LBM exhibits instability at high Reynolds number flows. Various methods have been published in the literature to mitigate this stability issue. Some approaches adopt the multi-relaxation-time method [58] or implement the cascaded LBM [59], both of which enhance the numerical stability of the LBM [33]. Other methods improved the BGK collision model using the Regularized BGK model, which recovers the macroscopic shear stress by maintaining the second order in the Knudsen number and assuming that the second-order non-equilibrium function can represent their highest order ($\epsilon f_i^{(1)} \approx f_i - f_i^{(0)}$) [60, 61]. The Regularized LBM shows a remarkable enhancement in stability and accuracy while only truncating up to the second-order non-equilibrium part, which is computationally inexpensive as it adds a few steps of computing the momentum flux tensor and the non-equilibrium distribution function [60]. Moreover, the Regularized LBM was improved into the recursive regularized LBM (RRLBM) by adding higher-order terms of the equilibrium and non-equilibrium particle distribution functions instead of the second-order in the regularized LBM, increasing stability and accuracy [62, 63]. Furthermore, other approaches increase the LBM's stability and accuracy by decoupling the numerical mesh from the lattice structure, allowing for independent space and time discretization [57]. This method is adopted for most LBM with unstructured mesh methods.

As the study aims to solve the flow around moving boundaries and reduce the computational resources, the flow solution must be modified due to the boundaries' presence and their motion. Thus, the two main challenges in implementing LBM are the boundary treatment

based on grid modification and moving boundary treatment. The boundary treatments aim to modify the solution due to the presence of the boundaries, as the uniform cartesian mesh does not precisely represent the actual boundaries. On the other hand, the moving boundary treatments aim to modify the solution due to the boundary motion. These two treatments are discussed in more detail in the following sections.

2.2 Boundary treatment based on grid modification

Utilizing a uniform mesh in the standard LBM imposes significant limitations on its applicability for complex boundaries. Thus, boundary treatments are required to increase the stability and accuracy of LBM. As discussed in Chapter 1, the boundary treatment could be classified into body fitting and orthogonal mesh refinement methods.

First, the body-fitting methods are the curvilinear mesh and the unstructured mesh methods. The body-conforming curvilinear mesh method [64, 65] gives accurate results but only applies to regular curvilinear boundaries such as circular ones. If the boundary is not near-circular, the mesh may be intersected or not uniform enough to adapt to the LBM.

The LBM unstructured meshing methods, such as finite-difference LBM [33, 66, 67], finite volume LBM [35], and finite element LBM [68] are based on the concept of independent space and time discretization LBM [57]. Moreover, a study introduced a moving boundary treatment with unstructured mesh by integrating ALE with FVLBM [37]. Furthermore, an unstructured grid based on the flux solver LBM [38] was introduced in the literature. It is based on interpolating macroscopic variables around the flux calculation point to construct the equilibrium PDFs needed for the flux solver. Besides confirming the boundaries, the unstructured mesh methods also perform grid refinement around the boundaries, giving accurate results as the unstructured mesh conforms to the non-regular boundaries. However, it adds more complex data structures, complex grid generation, and increased computational resources due to additional calculations, contradicting LBM's simplicity advantages on regular grids. Furthermore, some unstructured methods, such as FDLBM, could produce numerical instability at low relaxation time LBM, requiring a high-order discretization scheme [33].

Second, the orthogonal grid refinement methods could be classified into uniform and non-uniform orthogonal methods. The uniform orthogonal grid refinement methods, such as local grid refinement [39] and multi-block grid [25, 40], are based on performing the refinement with uniform

refinement blocks. The local grid refinement method introduces a grid refinement in locations of the most expected solution changes, which divides the mesh into two cells' blocks, fine and coarse cells. The fine-to-coarse displacement ratio n is not required to be two and can reach up to 50. As the grid size changes, the kinematic viscosity must be modified to retrieve the same viscosity and Reynolds number. Consequently, a new relaxation parameter must be calculated in the fine mesh by knowing the coarse mesh relaxation parameter as $\omega_f = 2/(1 + n(2/\omega_c - 1))$, where (ω_f, ω_c) are the fine and coarse grid relaxation parameters. However, the finer mesh performs the collision and streaming steps n times to maintain the same time step in both blocks. Moreover, special treatments on the PDFs transferred between coarse and fine cells must be performed.

Similarly, the multi-block grid method implements fine and coarse grids and calculates new relaxation parameters to maintain the same viscosity. Nevertheless, the coarse grid displacement is doubled as the finer one ($\Delta x_c = 2\Delta x_f$) [25], maintaining the same dynamic viscosity within the relation $(\tau_c - 0.5) \Delta x_c = (\tau_f - 0.5) \Delta x_f$. Consequently, these blocks could be repeated as needed to reach the required refinement and result in multi-blocks. Applying three blocks on a free fall cylinder test case decreases the processing time from 4494 seconds of a uniform grid to 227 seconds with a percentage of 1879% [25].

The non-uniform orthogonal grid refinement methods, such as the non-uniform staggered cartesian grid (NSCG) [44], quadtree LBM [41, 42], and non-conforming orthogonal mesh [43], are based on performing the refinement with a non-uniform orthogonal mesh near the boundaries. The NSCG method presents a nonuniform refinement based on introducing rectangular cells and has four types of cells: coarse, fine, vertical-fine, and horizontal-fine, and implements scaling on the kinematic viscosity for each cell. Consequently, an arithmetic average relaxation parameter was presented and applied to corner nodes of the vertical-fine and horizontal-fine cells [44]. This arithmetic average relaxation parameter allows all fluid points, whether coarse or fine, to be solved once each time step. However, it requires some treatments based on the refinement cell in all lattice directions. Moreover, when applied to 1-D refinement, it reduces the number of nodes compared to local grid refinement. Conversely, it increases the necessary number of nodes in 2D refinement, such as refinement around a cylinder [44].

The quadtree mesh is a data structure where the cells can be divided into four equivalent quadrants based on refinement requirements. These refinement requirements can be implemented

by adding control points in the domain. If a cell contains a control point, it is split into four other cells till it reaches the maximum required level l_{max} . A cell with no splitting has a level equal to zero; with each split, the new cells have their level increased by one. The algorithm of the LB quadtree method in literature [41, 42] is based on the local grid refinement method, where the calculation is performed between each quadtree's levels l . This reveals that the finest cells perform collision and streaming $2^{l_{max}}$ times the most coarse ones. This could lead to an extremely high processing time for high-level quadtree mesh. For instance, if the quadtree has $l_{max} = 6$, so the maximum level cells perform the collision and streaming $2^6 = 64$ times the low level cell. Moreover, the local grid refinement special treatment must be applied between each two consecutive level cells, adding more complexity and processing time.

Finally, the non-conforming orthogonal mesh method generates a quadtree-like mesh with large aspect ratio cells, as the x and y direction splitting are separated [43]. The core of the non-conforming LBM is using interpolated post-collision PDFs in the streaming step. These interpolated PDFs' locations are offset from the PDF pulling node with the smallest cell size dimensions, which leads to using the same time loop and relaxation time for all nodes in the non-conforming orthogonal mesh. However, it adds more complexity due to interpolation calculation.

Figure 2.3 shows the different boundary treatment methods based on the grid modification as a continuation of Figure 1.2. The curvilinear mesh method gives good results due to conforming the boundary; however, it has limited application because of its difficulty adapting to complex geometry. The unstructured mesh methods give accurate results due to boundary conforming and grid refinement; however, they add more complex data structures, complex grid generation, and increased computational resources. The uniform orthogonal grid refinement gives good results with the simplest way of refinement; however, the computational resources could be reduced by using the non-uniform refinement. Moreover, the interpolated streaming PDF scheme introduced in the non-conforming LBM could be more suitable with the quadtree mesh than the local refinement-based scheme used in the LB quadtree method.

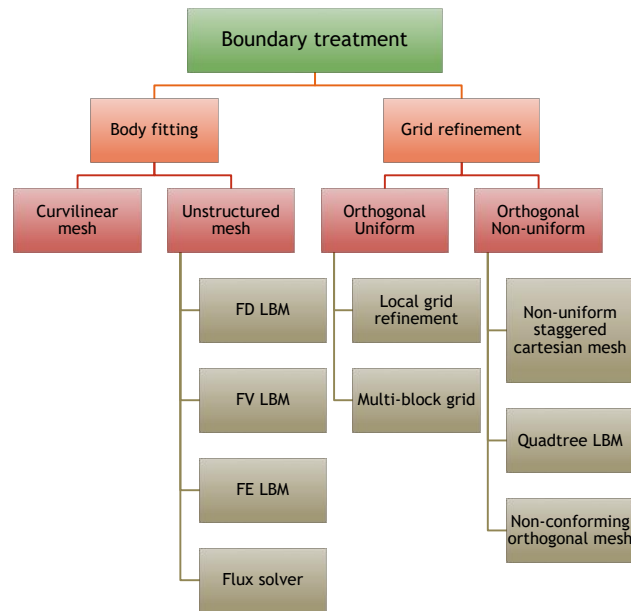


Figure 2.3 continues the boundary treatment hierarchy shown in Figure 1.2.

2.3 Moving boundary treatment

As discussed in Chapter 1, there are four methods to implement the moving boundary treatments with LBM: immersed boundary method, mass-conserved volumetric lattice Boltzmann method (MCVLBM), arbitrary Lagrangian-Eulerian (ALE) method, and LB-overset method, as shown in Figure 2.4.

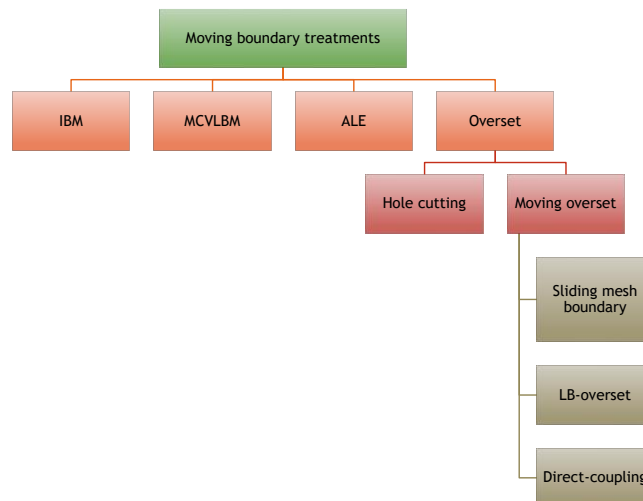


Figure 2.4 Moving boundary treatments hierarchy.

Firstly, the immersed boundary method was developed by Peskin [69] in the 1970s to simulate the blood flow in the heart [25, 45]. This method presents the fluid flow in an Eulerian cartesian mesh while the wall boundaries are immersed as arbitrary Lagrangian nodes, and both

nodes are presented in a single coordinate system, as shown in Figure 2.5. The body force is evaluated by spreading the Lagrangian forces from the immersed Lagrangian nodes into the ambient fluid nodes. This force spreading is widely performed using a smoothed approximation of the Dirac delta function [25, 50]. The LBM and IBM are based on the cartesian grid; thus, they are well suited to simulate the moving boundary treatments [25, 45, 70].

The idea behind the IB-LBM is to calculate a corrected body force, which results from immersing the boundaries (Lagrangian nodes) using an iterative process. Then, its effect is added to the fluid (Euler domain) by calculating the new particle distribution function using the multi-direct forcing method, where the no-slip boundary condition can be satisfied accurately [25]. The multi-direct forcing method is considered a bounceback improvement method that adds a source term containing the corrected body force to the post-collision particle distribution function (f_i^*), and the post-streaming PDF (f_i) is updated based on the corrected body force.

The IB-LBM is considered a straightforward method; however, it has a first-order accuracy in space due to the discontinuity of the velocity gradients on the boundaries. Nevertheless, this problem is addressed by proposing a higher-order IB-LBM with a more complex algorithm [71]. Moreover, IB-LBM has a jump effect, a discontinuity in stress balanced by forces existing between phases. This jump condition could be eliminated using the immersed interface method [72]; however, it increases the complexity of IB-LBM. Moreover, using many Lagrangian points adds complexity to implementing the IB-LBM and increases the required computational resources. As for each lagrangian point, a check of the neighboring points may be required to avoid intersecting or collisions, and add its effect in the Dirac smoothing function.

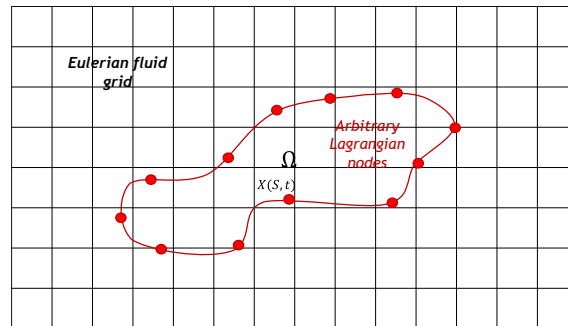


Figure 2.5 Illustration of the Eulerian fluid grid and the solid arbitrary lagrangian nodes of the IB-LBM.

Moreover, to obtain more accurate results regarding the boundary movement, grid refinement methods are required to be integrated with the IB-LBM [25]; thus, a multi-block

refinement method is suitable for IB-LBM as it also depends on the uniform cartesian mesh [73], as shown in Figure 2.6. However, it should be implemented as adaptive mesh refinement, adapted to the boundary motion.

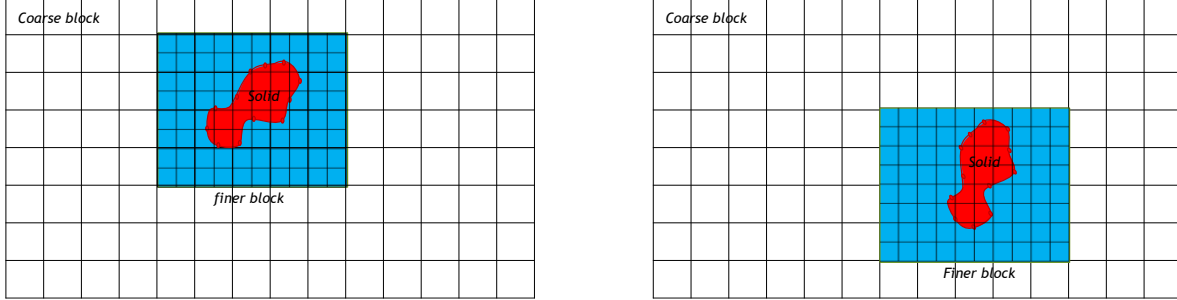


Figure 2.6 Multi-block movement in IB-LBM with the solid body motion presented for different times.

Furthermore, IB-LBM is widely used in deformable fluid-structure interaction (FSI) studies, such as the deformation of nanofluid droplets [50], fluid-particle interaction [45], and blood cell deformation through microchannels [49]. The arbitrary Lagrangian nodes can be deformed by adding its effect in the Euler grid through the deformation equation with the same Dirac smoothing function called the momentum exchange IB-LBM [50, 74]. Also, IB-LBM is applied for simulating two-phase flow with a large density ratio, such as a droplet moving in a constricted pipe. The high-density fluid is defined as the Lagrangian nodes, and the low-density fluid is defined as the Eulerian grid; thus, it has many advantages regarding the simplicity, high-density ratio fluids, and accuracy as some LBM two-phase flows cannot ensure the Galilean invariance and spurious velocities around the interface [25].

Secondly, the MCVLBM is developed for the FSI problems and moving boundary treatments. This method defines a volumetric parameter that represents the occupation of solid volume in the cells and classifies them into three types: solid, fluid, and boundary with P values of $(0, 1, 0 < P < 1)$, respectively [52]. Moreover, a volume-based lattice is proposed instead of a node-based lattice. The volume-based particle distribution function is uniformly distributed in each lattice cell/edge instead of the grid nodes in conventional LBM.

The boundary treatment applied in the MCVLBM is a modified streaming bounceback method applied to the volumetric concept that only reflects the particles' distribution functions. On the other hand, the moving boundary treatment is introduced through a boundary-induced

migration of particles to conserve the mass. The migration technique simultaneously evaluates the migrated particle distribution function using the volumetric parameter values and the streamed particle distribution functions [52].

The third method is the ALE-LBM method. This method presents the flow within two grids: a fixed grid, the reference frame, and the moving grid, which solves the flow with the effect of the moving boundary. The fluid data are stored and solved in both frames' nodes with respect to a single reference coordinate system, the fixed frame. Moreover, the communication between the two grids is implemented through interpolations; the communication from the moving grid to the fixed grid is implemented over the entire moving grid domain, while the communication from the fixed grid to the moving grid is implemented on the moving grid boundaries [53]. This could be considered a noticeable drawback due to the requirements of high computational resources. However, the post-collision PDF collision on the moving grid does not have to be corrected to consider the effects of relative accelerations between moving frames of reference, which is one of the ALE-LBM advantages: no additional source term exists to correct the collision operator as the system is described in a single Galilean invariant frame of reference. Adding the additional source term [54] could lead to inaccurate results when space acceleration is not uniform [53].

Moreover, a modified BGK model is introduced to solve the flow in both grids. This modification is presented in the microscopic velocity as it is redefined as a relative velocity ($\xi_i - \hat{u}$), where the variable ($\hat{u}$) is the moving frame velocity field. Moreover, a sequence of analytical derivations is developed by modifying the relaxation parameter on the discretized formulation of the modified BGK equation ($\omega = 1/(\tau + 0.5)$) instead of ($\omega = 1/\tau$), which leads to the ALE-LBM equation $f_i(\vec{x} + \xi_i \Delta t, t + \Delta t) = (1 - \omega)f_i(\vec{x} + \hat{u} \Delta t, t) + \omega f_i^{eq}(\vec{x} + \hat{u} \Delta t, t)$.

The boundary treatment applied in ALE is the non-equilibrium interpolation method combined with the modified BGK model, retrieving second-order accuracy in space. Also, Maldi et al. [53] retrieved the ALE Navier-Stokes equation from the ALE-LBM equation for low Mach number and mesh Mach number limit, maturing the ALE-LBM. Therefore, the ALE could be considered a medium-complex method with second-order accuracy. However, it can be applied only to the rigid moving boundaries and not to the FSI problems with non-uniform deformable boundaries, such as droplets moving in a constricted pipe, as the moving mesh shape is fixed and

the movement is restricted to the rigid boundaries. Moreover, it may consume a considerable processing time due to the interpolation step over the entire moving mesh.

Finally, the overset method is widely applied in computational fluid dynamics, especially with the Navier Stokes solvers [75-77]. It was first used in the 1980s in studying an airfoil by connecting a body-fixed O-type grid with a cartesian grid called the CHIMERA grid [78]. Later, many studies developed the overset method to solve the moving boundary problem with the hole-cutting approach [79, 80].

This approach has two overlapped grids: the refined grid conforming to the body wall and a course fixed grid. Those two grids communicate through interpolation on donor and receptor nodes on a fringe zone. Moreover, all the fixed grid nodes in the overlapped region are deactivated. So, the hole-cutting process refers to detecting the donor and receptor nodes and deactivating the fixed grid redundant nodes. However, this process must be evaluated with the refined grid motion at each time step, leading to a high processing time. Consequently, the moving overset grids method is developed where some interpolation or extrapolation techniques apply the connection between the moving and fixed grids without altering the grid structure [81].

The moving grid overset method is first introduced with the LBM solver as the sliding mesh interface method [54, 82, 83] because the domain is separated into two grids sharing only one interface ring, as shown in Figure 2.7(a). The interface ring aims to maintain the same fixed grid points during the rotation, and the moving grid outer boundary is selected as a circular pattern [54]. Moreover, using one interface could not lead to mass and momentum conservation and generate numerical noise due to the relative motion [53]. This issue is solved by introducing two communication borders, fixed-to-moving and moving-to-fixed, called the LB-overset method [55, 56], as shown in Figure 2.7(b). These communication borders act as boundary conditions; the fixed-to-moving border acts as a boundary condition for the moving grid, and the moving-to-fixed border acts as a boundary condition for the fixed grid [55].

The LB-overset method could be clearly defined as a second-order accuracy method that defines the domain as two grids with their respective coordinate systems; these grids are the fixed grid and the moving grid, and the solid body is attached to the latter. Both grids share an overlapping region and communicate through two interfaces using interpolations, which require

lower computational resources than ALE-LBM, as the interpolation is only performed on the borders instead of the entire domain.

The core of the LB-overset method is the reconstruction of PDFs on the communication borders, acting as boundary conditions for both grids. The PDFs' reconstruction is implemented by macroscopic interpolation [61] and microscopic interpolation, the so-called direct coupling method [56]. The macroscopic interpolation method interpolates the velocity, density, and flux tensor between the two grids on the borders. Then, the equilibrium and non-equilibrium PDFs are reconstructed from the interpolated macroscopic variables using RRLBM momentum equations. On the other hand, the microscopic interpolation performs the interpolation from the PDFs and then calculates the macroscopic variables from the interpolated PDFs to reconstruct the PDFs through their equilibrium and non-equilibrium parts. Moreover, it needs a rotation of the PDFs before the interpolation, which is a more complex process than the rotation of the macroscopic variables. Furthermore, it required interpolating the stress tensor required to reconstruct the non-equilibrium PDFs.

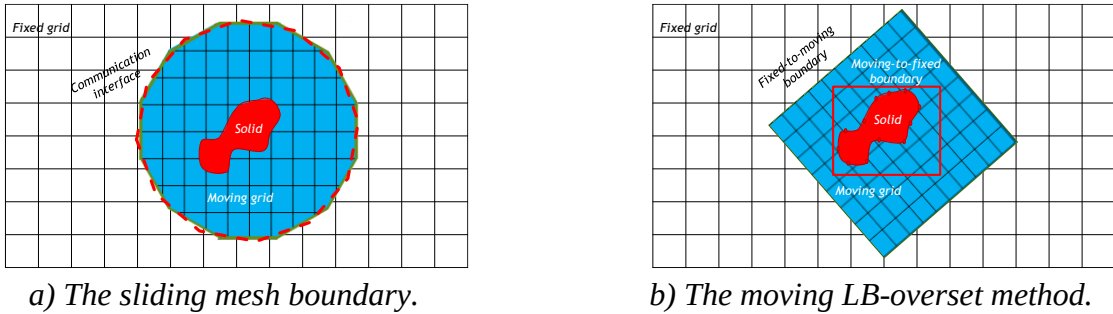


Figure 2.7. The difference in the grid communication between the sliding mesh boundary and the moving overset-LBM method.

Regarding the interpolation methods, most previous studies use the bilinear interpolator with nine surrounding nodes [55, 56]. These nine nodes are selected from the target node center, west, south, and southwest cells, which neglect the other surrounding cells. Furthermore, another study compares the bilinear interpolator and the gradient-based quadratic Hermite interpolation [61]. These interpolators use only the four surrounding points and have a second order of accuracy [61]. Moreover, the gradient-based interpolator is better than the bilinear interpolator as it suppresses the noise resulting from the border interpolations. However, the gradient-based interpolation is based on the strain tensor calculation, which may need further information from the surrounding cells in the case of single relaxation time (SRT) LBM, where the strain tensor is

not calculated. Thus, the gradient-based interpolator may be more suitable with the multi-relaxation time (MRT) LBM, where it was first introduced, as the strain tensor must be calculated [84].

Regarding the grid refinement on the LB-overset method, only two studies perform grid refinement on the LB-overset method. Based on the standard LB-overset method, one study applies a multi-block refinement only on the fixed grid, while the moving grid is a uniform cartesian grid with the finest cells [61]. The other one, based on the direct coupling LB-overset, applies the local grid refinement concept as the moving grid is a uniform cartesian grid with fine cells, while the fixed grid has coarse cells [56].

Both IB-LBM and MCVLBM have more complicated schemes than the ALE-LBM and LB-overset because of the adaptation of arbitrary boundary node motion. Moreover, they could consume more computational time than the latter, as these arbitrary nodes must be detected at the beginning of each time step. This arbitrary node detection is not required in ALE-LBM and LB-overset as the boundaries are attached to the moving grids. Thus, IB-LBM and MCVLBM are more suitable for the FSI problems, while ALE-LBM and LB-overset are more suitable for rigid boundary motion. Moreover, ALE-LBM and LB-overset have second-order accuracy in space and time. However, the ALE-LBM seems to have lower computational efficiency than the LB-overset as the ALE-LBM implements the interpolations over the moving grid's entire domain, compared with the interpolation on the borders in the LB-overset.

CHAPTER 3 RESEARCH GAP, OBJECTIVES, AND OUTLINE

This chapter illustrates the research gap based on the previous literature survey. Then, the study objectives required for filling the gap are explained. Finally, the study outlines based on the objectives are stated.

3.1 Research Gap

Based on the literature review, the LB-overset is the most suitable moving boundary treatment method with the study goal as it solves the moving rigid boundaries while maintaining the LBM governing equations, besides retrieving a second order of accuracy in space and time. Only two studies present grid refinement on the LB-overset method; one applied multi-block refinement on the fixed grid only [61], while the other applied the local refinement concept as the whole moving grid has fine uniform cartesian grids [56]. None of those studies dealt with grid refinement around the boundaries. Moreover, no publication studied the LB-overset method with the quadtree grid refinement based on either the local refinement approach or the interpolated PDF streaming approach, revealing the research gap.

The primary research gap could be stated: ***“No study has been done on the LB-overset method with quadtree grid refinement based on either the local refinement approach or the interpolated PDF streaming approach.”***

Moreover, the literature review reveals few studies that discuss the interpolation methods applied to the LB-overset grid communication. One compares only two interpolation methods, revealing a minor research gap regarding *investigating more interpolation methods and their effect on the LB-overset method*. Furthermore, the quadtree grid refinement method does not conform to the boundary; thus, a boundary treatment for curved boundaries must be integrated into the overall system. Figure 3.1 shows the literature survey's overall picture and the research gap, leading to revealing the research objectives in the following section.

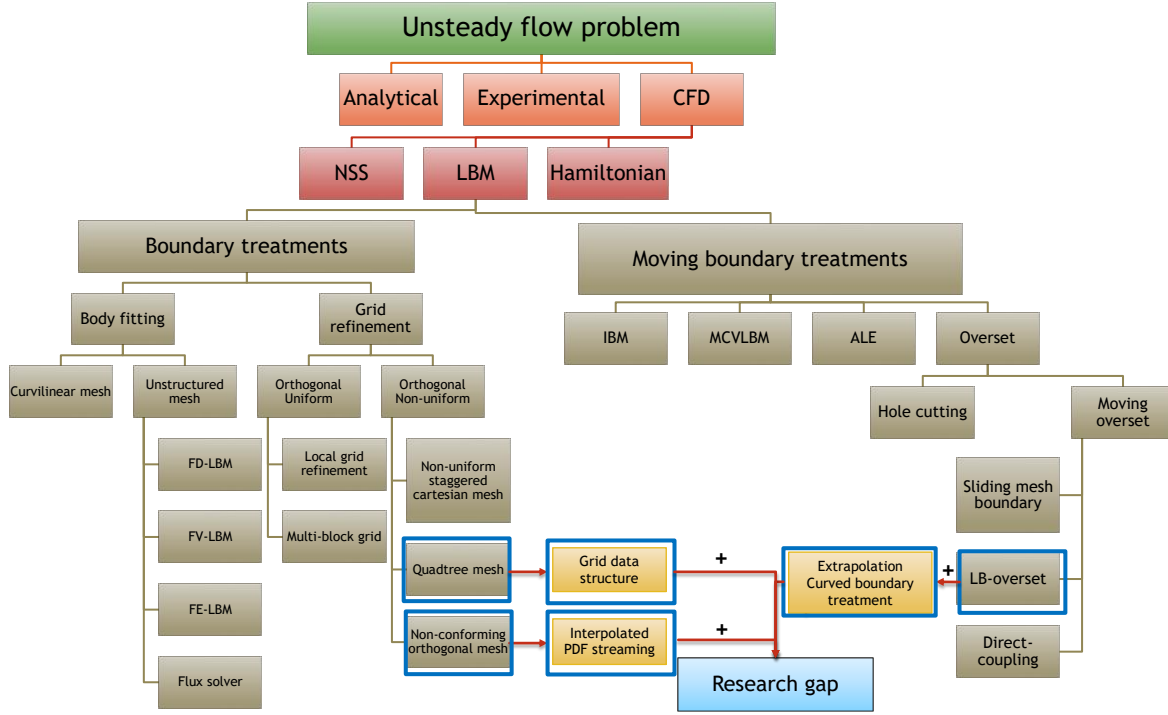


Figure 3.1. The literature survey's overall picture and the research gap.

3.2 Objectives

The study objectives are based on the research gaps, and they are classified into the main objective and sub-objectives. The main objective is the work that fills the research gap while satisfying the sub-objectives leads to achieving the main objective, and they can be articulated as follows:

1- The main objective is to **verify and code-to-code verify the integrated Lattice Boltzmann method with the moving overset approach, coupled with quadtree grid refinement based on the interpolated PDF streaming scheme.**

2- The sub-objectives are stated as follows:

- a) Verify and code-to-code verify the standard LBM with linear and curved boundary treatments. The verification and code-to-code verification will be implemented using test cases: Poiseuille flow, Taylor-Couette flow, flow around stationary cylinder, and flow around rotating cylinder.
- b) Verify and code-to-code verify the LBM based on the recursive regularized BGK model with the overset method for solving the moving boundary problem.

- c) Investigate the different interpolation methods and their influence on the LB-overset grids communications. The verification, code-to-code verification, and interpolators comparison will be achieved using test cases: uniform flow, Poiseuille flow, Taylor-Couette flow, high Reynolds advection vortex in uniform flow, and flow around a rotating cylinder.
- d) Design a quadtree mesh generation algorithm and its corresponding data structure. Then, implement the interpolated PDF streaming process.
- e) Integrate the quadtree mesh refinement (d) to the LB-overset method (b), then verify and validate the integrated method to reach the primary objective. The verification and code-to-code verification will be implemented using test cases: uniform flow, Poiseuille flow, Taylor-Couette flow, and flow around a rotating cylinder.

3.3 Outlines

The technical part of the study is presented in three chapters, achieving the objectives as follows:

Chapter 4: This Chapter discusses the solution of the LBM with curved boundary, describing the standard LBM algorithm and its boundary conditions, the mesh generation algorithm and its data structure, and the boundary nodes detection algorithm, which covers the sub-objective (2. a).

Chapter 5: This Chapter is a submitted manuscript titled “*Impact of interpolation method selection on the accuracy of the lattice Boltzmann-overset method, Computers & Mathematics with Applications*”. It studies the integration overset method with the recursive regularized LBM using different interpolation schemes, giving a complete description of the algorithm and the interpolation schemes. Moreover, steady and unsteady test cases are implemented to compare the interpolation schemes and method’s verification and code-to-code verification, achieving the sub-objectives (2. b) and (2. c).

Chapter 6: This Chapter is a submitted manuscript titled “*Integration of Lattice Boltzmann-overset method with non-conforming quadtree mesh based on the combination of spatial and Lagrangian-link interpolated streaming technique, Journal of Computational Science*”. It studies integrating the RRLB-overset method with the quadtree mesh refinement based on a developed interpolated PDF streaming scheme, which satisfies the sub-objective (2. d) and (2. e). The interpolated PDF streaming scheme is a spatial Lagrangian-link interpolation method that enables the use of the same

time step, viscosity, and relaxation time for both grids' nodes. Furthermore, it minimizes the nodes required for interpolation, reducing computational resource demands and enhancing simulation efficiency.

Following these chapters, the general discussion and the thesis conclusion are discussed in Chapter 7 and Chapter 8, respectively.

CHAPTER 4 TECHNICAL CHAPTER

This chapter aims to achieve the first sub-objective of validating and verifying the standard LBM with Guo's extrapolation method for solving the curved boundaries. First, in section 4.1, the standard LBM is discussed with the implementation procedures. Then, the boundary conditions used in this study are introduced in section 4.2. These boundary conditions are the halfway bounce-back, the straight velocity boundary condition, the Zou-He inlet and outlet boundary conditions, and the extrapolation method for curved boundary treatment. After that, In section 4.3, the mesh generation is illustrated by defining the data structure and the mesh generation algorithm. Moreover, section 4.4 illustrates the procedures for implementing the LBM with the extrapolation method. Finally, the results and discussion are presented in section 4.5.

4.1 Standard LBM

This section discusses the implementation of the standard D2Q9 LBM with the BGK collision model. For more details on LBM, the reader is referred to section 2.1. The lattice Boltzmann equation with no external force can be written as

$$f_i(\mathbf{x} + \xi_i \Delta t, t + \Delta t) - f_i(\mathbf{x}, t) = \Omega_i(\mathbf{x}, t) \Delta t \quad (4.1)$$

Where $\Omega_i(\mathbf{x}, t)$ is the collision operator, and Δt is the time step. This collision operator is a complex double integral over the velocity spaces, considering all the possible outcomes of two-particle collisions for intermolecular forces [28]. The BGK model defines the collision operator with a single relaxation of the distribution function to its local equilibrium distribution $f_i^{eq}(\mathbf{x}, t)$. The collision and streaming steps using the BGK collision model can be defined in equations (4.2) and (4.3).

$$f_i^*(\mathbf{x}, t) - f_i(\mathbf{x}, t) = -\frac{1}{\tau} \left(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t) \right) \quad (4.2)$$

$$f_i(\mathbf{x} + \xi_i \Delta t, t + \Delta t) = f_i^*(\mathbf{x}, t) \quad (4.3)$$

Where τ is the relaxation time, the post-collision particle distribution function is $f_i^*(\mathbf{x}, t)$, and the post-streaming particle distribution function is $f_i(\mathbf{x} + \xi_i \Delta t, t + \Delta t)$. Furthermore, the equilibrium distribution can be defined as

$$f_i^{eq}(\mathbf{x}, t) = w_i \rho \left(1 + \frac{\xi_i \cdot \mathbf{u}}{c_s^2} + \frac{(\xi_i \cdot \mathbf{u})^2}{2 c_s^4} - \frac{\mathbf{u} \cdot \mathbf{u}}{2 c_s^2} \right) \quad (4.4)$$

Where the lattice speed of sound is c_s , the lattice weights is w_i . The density ρ and velocity $\mathbf{u}$ can be calculated from the zeroth, and the first moments of the particle distribution function for the D2Q9 model as

$$\rho = \sum_{i=0}^8 f_i \quad (4.5)$$

$$\rho \mathbf{u} = \sum_{i=0}^8 \xi_i f_i \quad (4.6)$$

Figure 4.1 shows the flow chart of the standard LBM, which could be mentioned in the following procedures:

- a) Define the flow variables and the initial conditions.
- b) Start the time loop, then apply the inlet and outlet boundary conditions, such as the Zou and He boundary condition.
- c) Calculate the equilibrium particle distribution function $f_i^{eq}(\mathbf{x}, t)$ using equation (4.4).
- d) Apply the collision by calculating the post-collision particle distribution function $f_i^*(\mathbf{x}, t)$ in equation (4.2).
- e) Apply the streaming through equation (4.3) to evaluate the post-streaming particle distribution function $f_i(\mathbf{x} + \xi_i \Delta t, t + \Delta t)$.
- f) Apply the boundary conditions that modify the post-streaming particle distribution function through the post-collision one, such as the bounce-back or Inamuro slip boundary conditions.
- g) Update the macroscopic variables using equations (4.5) and (4.6).
- h) Repeat the procedures from c) to g) until the final simulation time.

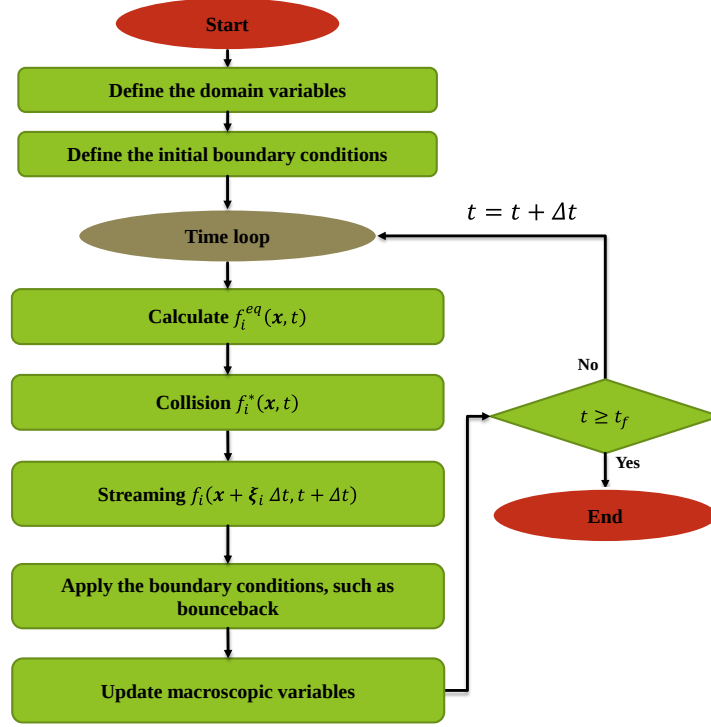


Figure 4.1. The Flow chart of the standard LBM.

The flow chart includes the implementation of the boundary conditions, which are illustrated in the following section.

4.2 Boundary conditions

This section illustrates the boundary conditions used in this study. First, the Zou and He velocity inlet/pressure outlet boundary conditions [85] are discussed in section 4.2.1. Then, the straight velocity boundary condition introduced by Inamuro is discussed in section 4.2.2. After that, in section 4.2.3, the halfway bounce-back implementation is discussed. Finally, the curved boundary treatment using the extrapolation method is presented in section 4.2.4.

4.2.1 Zou and He velocity inlet/ pressure outlet

The Zou and He velocity inlet/pressure outlet boundary conditions [85] construct the unknown post-streaming particle distribution function on the inlet and outlet boundaries for tangential velocities equal to zero $u_{in}^{\parallel} = u_{out}^{\parallel} = 0$. Figure 4.2. shows the unknown post-streaming particle distribution function as dashed red arrows. For example, the inlet boundary nodes have unknown values for f_1 , f_5 , and f_8 . These values are reconstructed from the imposed perpendicular velocity

profile $u_{\perp}$ on the inlet, the reconstructed density ρ_{in} , and the known post-collision particle distribution function as follows:

$$\rho_{in} = \frac{[f_0 + f_2 + f_4 + 2(f_3 + f_6 + f_7)]}{1 - u_{in}^{\perp}} \quad (4.7)$$

$$f_1 = f_3 + \frac{2}{3} \rho_{in} u_{in}^{\perp} \quad (4.8)$$

$$f_5 = f_7 - \frac{1}{2} (f_2 - f_4) + \frac{1}{6} \rho_{in} u_{in}^{\perp} \quad (4.9)$$

$$f_8 = f_6 + \frac{1}{2} (f_2 - f_4) + \frac{1}{6} \rho_{in} u_{in}^{\perp} \quad (4.10)$$

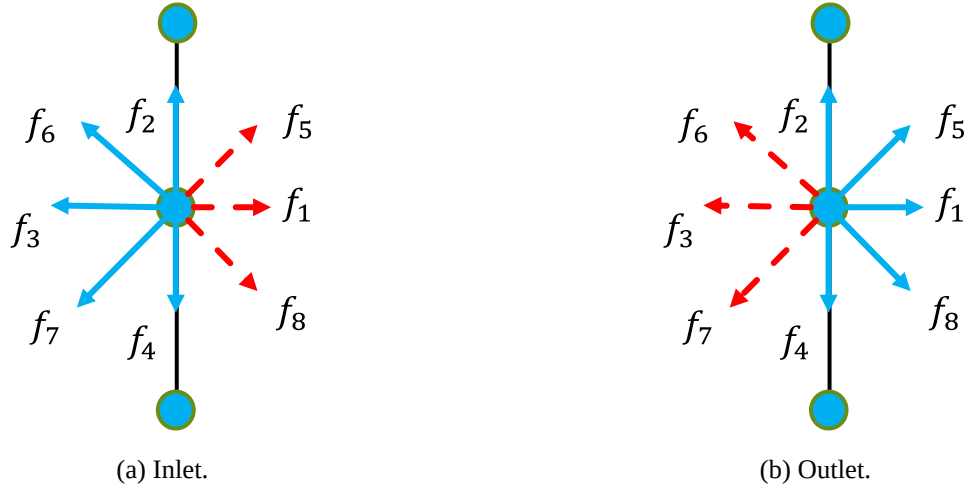


Figure 4.2. The unknown post-streaming particle distribution function in the inlet and outlet boundaries. The unknown values are shown in dashed red arrows, while the known from the streaming step are shown in solid blue arrows.

On the other hand, the outlet boundary nodes have unknown values for f_3 , f_6 , and f_7 , which are reconstructed from the imposed density (pressure) value ρ on the outlet boundary, the reconstructed perpendicular velocity $u_{\perp}$, and the known post-collision particle distribution function as follows:

$$u_{out}^{\perp} = 1 + \frac{[f_0 + f_2 + f_4 + 2(f_1 + f_5 + f_8)]}{\rho_{out}} \quad (4.11)$$

$$f_3 = f_1 + \frac{2}{3} \rho_{out} u_{out}^{\perp} \quad (4.12)$$

$$f_7 = f_5 - \frac{1}{2}(f_4 - f_2) + \frac{1}{6} \rho_{out} u_{out}^\perp \quad (4.13)$$

$$f_6 = f_8 + \frac{1}{2}(f_4 - f_2) + \frac{1}{6} \rho_{out} u_{out}^\perp \quad (4.14)$$

It is essential to state that the Zou and He boundary conditions are only suitable for low Reynolds numbers due to instability behavior at high Reynolds numbers [86]. Moreover, the velocity inlet and outlet profile must be perpendicular to the boundaries $u_{in}^\parallel = u_{out}^\parallel = 0$.

4.2.2 Straight velocity boundary conditions (Inamuro)

The Inamuro boundary condition [87] implements the velocity boundary condition on straight boundaries. It calculates the unknown particle distribution function based on defining a fictitious velocity $\mathbf{u}' = \mathbf{u} + \mathbf{s}$ and density ρ' . Where $\mathbf{s}$ is the velocity slip tangential to the wall ($s_\parallel = 0$), and the velocity field $\mathbf{u}$ is imposed from equation (4.6). Figure 4.3 shows the upper boundary's unknown post-streaming particle distribution function f_4 , f_7 , and f_8 as dashed red arrows.

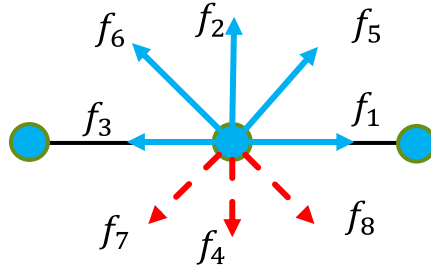


Figure 4.3. The unknown post-streaming particle distribution function in the upper boundary.

These unknown particle distribution functions are reconstructed using the equilibrium part constructed using the fictitious velocity and density as

$$f_i = f_i^{eq}(\rho', u_\perp + s_\perp, u_\parallel) \text{ for } i = 4, 7, 8 \quad (4.15)$$

The fictitious density and the slip velocity are calculated from the following equations:

$$\rho' = 6 \frac{-\rho u_\parallel + f_5 + f_2 + f_6}{1 + 3u_\parallel^2 - 3u_\parallel} \quad (4.16)$$

$$s_\perp = -u_\perp + 6 \frac{\rho u_\perp + f_3 + f_6 - f_5 - f_1}{\rho'(1 - 3u_\parallel)} \quad (4.17)$$

The Inamuro boundary is a good choice for laminar flow as it gives a high accuracy and is suitable for low Reynold numbers [86].

4.2.3 Halfway bounce-back

There are two types of bounceback: full-way and half-way bounceback. The full-way bounceback reflects the particle distribution function on the next collision step, as the particles move the whole way from the boundary to the solid nodes, as shown in Figure 4.4.a. On the other hand, the half-way bounceback reflects the particle distribution function after the current collision step and before the current streaming step, which means that the particles move only halfway and then are reflected at half of the time step, as shown in Figure 4.4.b. Moreover, both approaches have the physical wall on half the spatial displacement away from the solid node to ensure second-order accuracy in space.

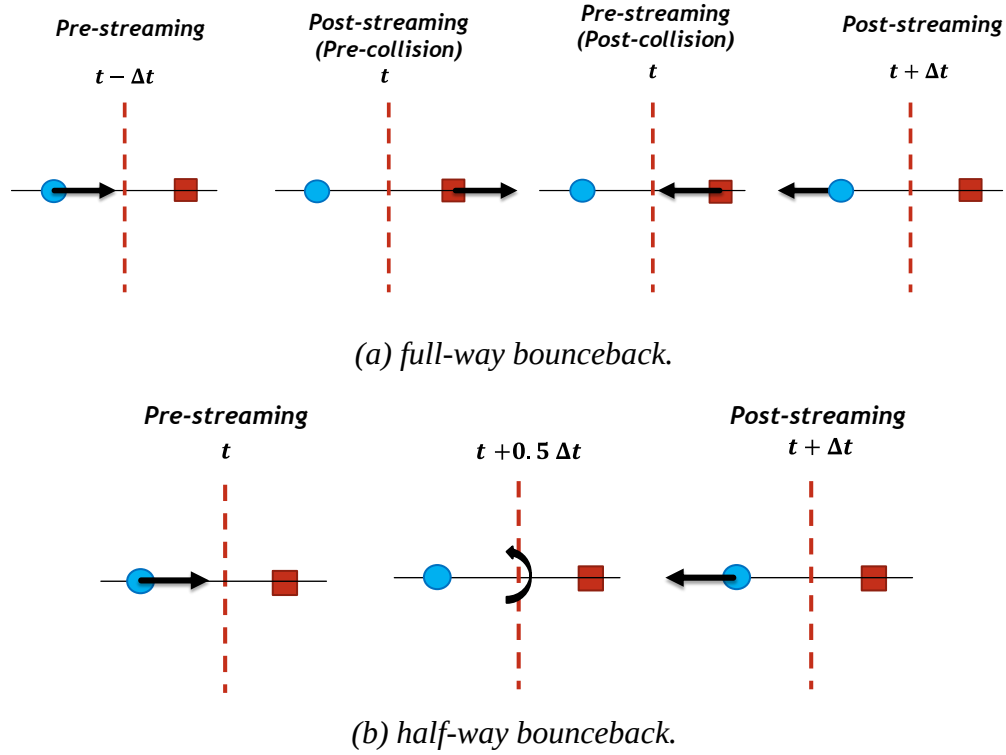


Figure 4.4. The timing difference between the full-way and half-way bounceback schemes. The fluid nodes are presented as circles, the solid nodes are presented as squares, and the physical wall is presented as a dashed line.

Furthermore, the full-way bounceback requires creating and saving the PDFs at the solid nodes where they are set for a time step. On the contrary, the half-way bounceback did not require the solid nodes, as the particles never reached these nodes. Moreover, the halfway bounceback is preferable for the transient problems as the halfway requires only Δt to reflect the particle

distribution function, while the full-way requires $2\Delta t$, which degrades the time accuracy in transient problems [28]. Thus, the half-way bounceback is more widely used than the full-way bounceback.

The halfway bounce-back is a no-slip boundary condition that defines a straight-line physical wall. The physical wall is applied on half the spatial displacement away from the bounce-back boundary, as shown in Figure 4.5. Thus, the computational domain starts from 0.5 to $N_y + 0.5$ with a domain height of N_y , where N_y is the cells' number in the y-direction. On the other hand, the channel started from 0 to $N_y + 1.0$ with height of $N_y + 1$.

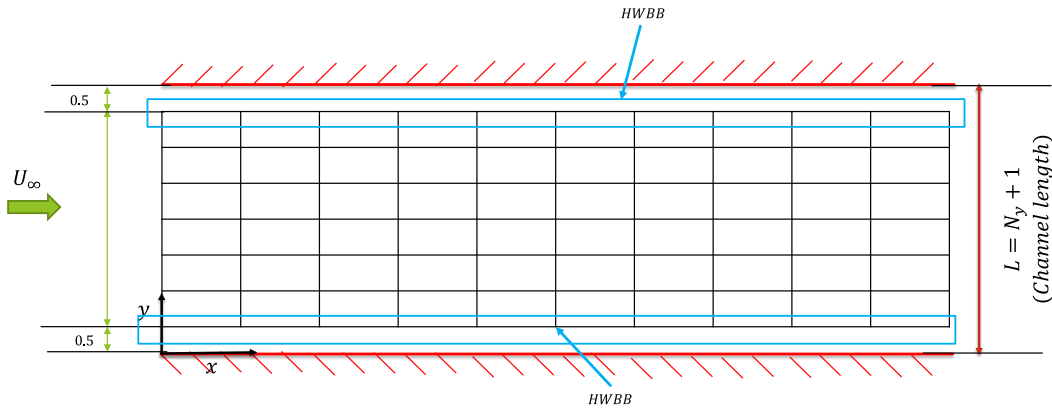


Figure 4.5. Schematic showing the location of the wall at the half of spatial displacement on a node-base computational domain for the halfway bounce-back boundary condition.

The bounce-back method redefines the unknown particle distribution function through the post-collision particle distribution function of the opposite direction at the same node. For example, the upper wall nodes have unknown post-streaming particle distribution functions of f_4 , f_7 , and f_8 . These post-streaming particle distribution functions are redefined as $f_4 = f_2^*$, $f_7 = f_5^*$, and $f_8 = f_6^*$. It has a spatial second-order accuracy for straight boundaries but a first-order accuracy for curved boundaries.

4.2.4 Curved boundary treatment (Guo's extrapolation method)

The Guo's extrapolation method [30] retrieves second-order spatial accuracy for the curved boundary by reconstructing the pre-collision particle distribution functions on the boundary nodes. The reconstruction is implemented using the first-order Chapman–Enskog expansion as

$$f_i = f_i^{eq} + f_i^{Neq} \quad (4.18)$$

First, the equilibrium PDF is constructed using the macroscopic variables $(\mathbf{u}, \rho)$ in equation (4.4). However, these variables are unknown on the boundary nodes; thus, extrapolation is performed from the neighboring fluid nodes to the boundary nodes to calculate these values as

$$\rho(\mathbf{x}_B) = \rho(\mathbf{x}_f) \quad (4.19)$$

$$\mathbf{u}(\mathbf{x}_B) = \frac{\mathbf{u}_w + (\Delta - 1) \mathbf{u}(\mathbf{x}_f)}{\Delta}, \Delta \geq 0.75 \quad (4.20)$$

$$\mathbf{u}(\mathbf{x}_B) = \mathbf{u}_w + (\Delta - 1) \mathbf{u}(\mathbf{x}_f) + (1 - \Delta) \frac{(2 \mathbf{u}_w + (\Delta - 1) \mathbf{u}(\mathbf{x}_{ff}))}{(1 + \Delta)}, \Delta < 0.75 \quad (4.21)$$

Figure 4.6 shows the different locations for a specific direction and a boundary node $\mathbf{x}_B$, neighboring fluid node $\mathbf{x}_f$, the following neighboring fluid node $\mathbf{x}_{ff}$, the imposed wall velocity $\mathbf{u}_w$, and the displacement function Δ .

$$\Delta = \frac{\|\mathbf{x}_w - \mathbf{x}_B\|}{\|\mathbf{x}_B - \mathbf{x}_f\|} \quad (4.22)$$

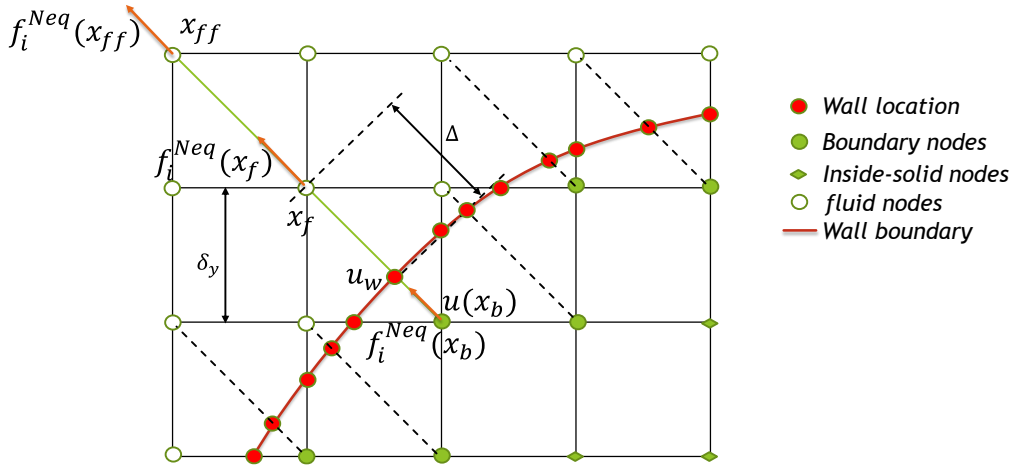


Figure 4.6. Schematic showing extrapolation technique different locations and their corresponding variables.

Similarly, the nonequilibrium PDF is calculated by extrapolating the nonequilibrium PDFs of the neighboring fluid nodes as:

$$f_i^{ne}(\mathbf{x}_b, t) = f_i^{ne}(\mathbf{x}_f, t) \quad , \Delta \geq 0.75 \quad (4.23)$$

$$f_i^{ne}(\mathbf{x}_b) = \Delta f_i^{ne}(\mathbf{x}_f) + (1 - \Delta) f_i^{ne}(\mathbf{x}_{ff}) \quad , \Delta < 0.75 \quad (4.24)$$

Then, the post-collision particle distribution function can be calculated by the constructed pre-collision particle distribution function in equation (4.18) using equation (4.2) or by using the equilibrium and the nonequilibrium parts as:

$$f_i^*(\mathbf{x}, t) = f_i^{eq}(\mathbf{x}, t) + \left(1 - \frac{1}{\tau}\right) f_i^{Neq}(\mathbf{x}, t) \quad (4.25)$$

Furthermore, two conditional arrays are calculated to classify each boundary node with its corresponding extrapolation equation. The conditional arrays are used to avoid if-statements inside the time loop as the code is vectorized. The conditional arrays are:

$$CA_{1\Delta}(\mathbf{x}_b) = \begin{cases} 1 & \text{for } 1 > \Delta \geq 0.75 \\ 0 & \text{otherwise} \end{cases} \quad (4.26)$$

$$CA_{2\Delta}(\mathbf{x}_b) = \begin{cases} 1 & \text{for } 0.75 > \Delta > 0 \\ 0 & \text{otherwise} \end{cases} \quad (4.27)$$

Consequently, equations (4.20) to (4.24) can be rewritten as:

$$\begin{aligned} \mathbf{u}(\mathbf{x}_b, t) = & \left(\frac{\mathbf{u}_w + (\Delta - 1) \mathbf{u}(\mathbf{x}_f)}{\Delta} \right) CA_{1\Delta}(\mathbf{x}_b) \\ & + \left(\mathbf{u}_w + (\Delta - 1) \mathbf{u}(\mathbf{x}_f) + (1 - \Delta) \frac{(2 \mathbf{u}_w + (\Delta - 1) \mathbf{u}(\mathbf{x}_{ff}))}{(1 + \Delta)} \right) CA_{2\Delta}(\mathbf{x}_b) \end{aligned} \quad (4.28)$$

$$f_i^{ne}(\mathbf{x}_b, t) = (f_i^{ne}(\mathbf{x}_f, t)) CA_{1\Delta}(\mathbf{x}_b) + (\Delta f_i^{ne}(\mathbf{x}_f) + (1 - \Delta) f_i^{ne}(\mathbf{x}_{ff})) CA_{2\Delta}(\mathbf{x}_b) \quad (4.29)$$

Now, it is crucial to know how to detect the boundary, fluid, and solid nodes and calculate the displacement function, which is introduced in the following section.

4.3 Mesh generation

This section discusses the mesh data structure, the mesh generation algorithm, and its procedures. First, section 4.3.1 describes the developed data structure defining the two main arrays: the node array and the surrounding array. Then, in section 4.3.2, the mesh generation algorithm is illustrated with its three main parts: the solid node detection algorithm, the boundary nodes

detection algorithm, and the displacement function calculations. Finally, the mesh generation procedures and its flow chart are discussed in section 4.3.6.

4.3.1 Data structure

The data structure is based on the required data for implementing the curved boundary treatment using the extrapolation technique. Two arrays are needed for describing the mesh: the nodes array and the surrounding array. The nodes array is based on the cell index, which retrieves the global node index of the four nodes forming a specific cell. This array is required for solid, fluid, and boundary nodes detection. The definition of the nodes array is as follows:

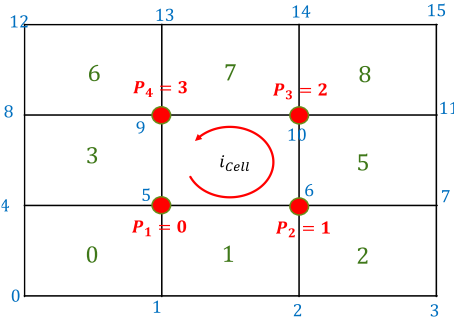
$$Node(P_1) = Node(i_{cell}, 0) = i_{cell} + i_y \quad (4.30)$$

$$Node(P_2) = Node(i_{cell}, 1) = i_{cell} + i_y + 1 \quad (4.31)$$

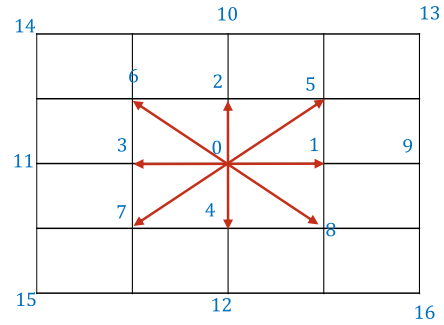
$$Node(P_3) = Node(i_{cell}, 2) = i_{cell} + i_y + n_x + 1 \quad (4.32)$$

$$Node(P_4) = Node(i_{cell}, 3) = i_{cell} + i_y + n_x \quad (4.33)$$

Where i_{cell} is the cell index, the row index measured at P_1 is i_y . For instance, in Figure 4.7 (a), $i_{cell} = 4$ and $i_y = 1$; thus, the nodes array retrieves the values of 5, 6, 9, and 10 using equations (4.30) to (4.33).



(a) Nodes array.



(b) Surrounding array.

Figure 4.7. Schematic showing the indices and directions for the Nodes array and Surrounding array.

On the other hand, the surrounding array is based on the global node index, which defines the neighboring and the following neighboring nodes. This array is required for retrieving the neighboring macroscopic variables and the non-equilibrium particle distribution function needed to implement the extrapolation method. The surrounding array is defined as follows:

$$Sur(i_{Node}, 0) = i_{Node} \quad (4.34)$$

$$Sur(i_{Node}, 1) = i_{Node} + 1 \quad (4.35)$$

$$Sur(i_{Node}, 2) = i_{Node} + n_x \quad (4.36)$$

$$Sur(i_{Node}, 3) = i_{Node} - 1 \quad (4.37)$$

$$Sur(i_{Node}, 4) = i_{Node} - n_x \quad (4.38)$$

$$Sur(i_{Node}, 5) = i_{Node} + n_x + 1 \quad (4.39)$$

$$Sur(i_{Node}, 6) = i_{Node} + n_x - 1 \quad (4.40)$$

$$Sur(i_{Node}, 7) = i_{Node} - n_x - 1 \quad (4.41)$$

$$Sur(i_{Node}, 8) = i_{Node} - n_x + 1 \quad (4.42)$$

$$Sur(i_{Node}, 9) = i_{Node} + 2 \quad (4.43)$$

$$Sur(i_{Node}, 10) = i_{Node} + 2 n_x \quad (4.44)$$

$$Sur(i_{Node}, 11) = i_{Node} - 2 \quad (4.45)$$

$$Sur(i_{Node}, 12) = i_{Node} - 2 n_x \quad (4.46)$$

$$Sur(i_{Node}, 13) = i_{Node} + 2(n_x + 1) \quad (4.47)$$

$$Sur(i_{Node}, 14) = i_{Node} + 2(n_x - 1) \quad (4.48)$$

$$Sur(i_{Node}, 15) = i_{Node} - 2(n_x + 1) \quad (4.49)$$

$$Sur(i_{Node}, 16) = i_{Node} - 2(n_x - 1) \quad (4.50)$$

Where i_{Node} is the global node index. Moreover, adding 8 to the neighboring indices in the same direction defines the next-neighbor nodes indeces. For example, in Figure 4.7 (b), the northeast direction has a neighboring index of 6; thus, the next-neighbor index is $6 + 8 = 14$. This constant addition gives a more convenient coding technique as the extrapolation equations (1) and (2) only require the data of the neighboring and next-neighboring in the same direction. For instance, the extrapolated data $f_{Neq}(0,6)$ is based on $f_{Neq}(Sur(0,6), 6)$ and $f_{Neq}(Sur(0,6+8), 6)$, globally, the equation (4.24) is defined as

$$f_i^{ne}(\vec{x}_b, i) = \Delta f_i^{ne}(Sur(\vec{x}_b, i), i) + (1 - \Delta) f_i^{ne}(Sur(\vec{x}_b, 2i), i), \quad i = 1, 2, \dots, 8 \quad (4.51)$$

4.3.2 Mesh generation algorithm

The Mesh generation algorithm is divided into three main parts: detecting the solid nodes, detecting boundary cells and nodes, and calculating the displacement function.

4.3.3 Detect solid nodes

First, the solid node array $S_N(\mathbf{x}_p)$ is created to contain a value equal to one if the node is a solid node and zero otherwise. Detecting the solid nodes is implemented using the winding number algorithm [88]. The winding number algorithm detects if the nodes are inside a polygon or not based on an infinite horizontal ray originating from the node. When this array intersects with a polygon segment, a local number is counted as +1 or -1, based on the segment direction, while it crosses the ray. When the direction is upward, an increment +1 is added, while a decrement (-1) is applied when it is downward. Then, the sum of these local numbers gives the winding number; if it equals the number of the polygon segments n_{seg} , the node is inside the polygon, as shown in Figure 4.8.

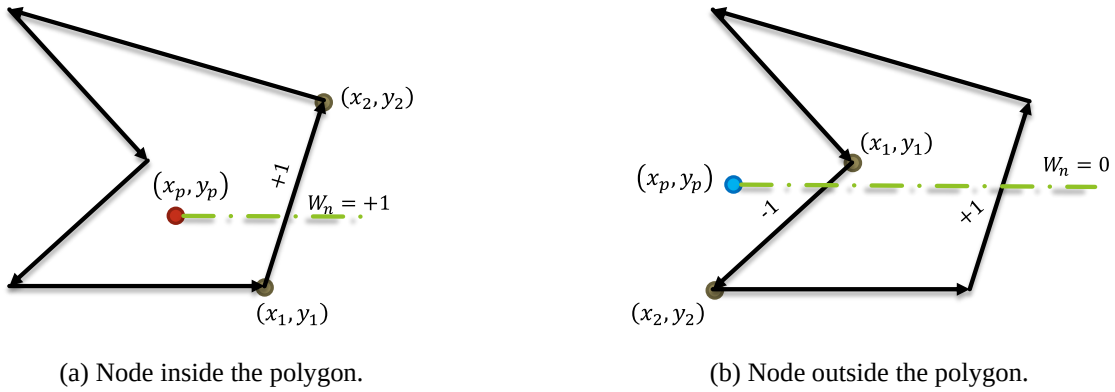


Figure 4.8. Schematic showing winding number algorithm.

The winding number algorithm is implemented using a test function $Test$, shown in equation (4.52), and three conditional arrays as:

$$Test = (y_p - y_1)(x_2 - x_1) - (x_p - x_1)(y_2 - y_1) \quad (4.52)$$

$$CA(x_p, i_{seg}) = \begin{cases} 1 & Test > 0 & \text{Point on segment's left} \\ -1 & Test < 0 & \text{Point on segment's right} \\ 0 & Test = 0 & \text{Point on segment} \end{cases} \quad (4.53)$$

Test array detects the point's location on the right of, left of, or on the line segment. *CA* is the conditional array stores a plus, minus, or zero of the point regarding each segment based on the test function. Then, the winding number array $W_n(\mathbf{x}_p)$ equals the sum of the conditional array elements $CA(\mathbf{x}_p, i_{seg})$ in the segment index dimension, which detects if the point is inside or outside the polygon. If the sum equals the number of segments n_{seg} , it is inside the polygon; otherwise, it is outside.

$$W_n(\mathbf{x}_p) = \sum_{i_{seg}=0}^{i_{seg}=n_{seg}-1} CA(\mathbf{x}_p) \quad (4.54)$$

Moreover, in case $Test = 0$, the point is located exactly on the segment or its extension; thus, another conditional array $CA_2(\mathbf{x}_p)$ is applied in the case of $Test = 0$ to ensure that the point lies on the segment and not its extension in the y-direction as:

$$CA_2(\mathbf{x}_p) = (y_p \leq y_2)(y_p \geq y_1)(y_2 \geq y_1) + (y_p \geq y_2)(y_p \leq y_1)(y_2 \leq y_1) \quad (4.55)$$

The solid node array S_N is defined as 1 for all nodes inside the wall boundary; otherwise, it equals zero. Furthermore, in the case of $Test = 0$, the points are considered solid nodes because they lie on boundary segments, so an additional condition is added to define the solid nodes by adding these nodes directly after the testing in equation (4.53):

$$S_N(\mathbf{x}_p) = \begin{cases} 1 & (W_n = n_{seg}) \\ 1 & (Test = 0) \cdot CA_2(\mathbf{x}_p) \\ 0 & W_n \neq n_{seg} \end{cases} \quad (4.56)$$

Moreover, like the solid node array, a fluid node array $F_N(\mathbf{x}_p)$ is defined to contain a value of one if the node is fluid and zero otherwise $F_N(\mathbf{x}_p) = 1 - S_N(\mathbf{x}_p)$. The solid and fluid node arrays are essential for detecting boundary cells and nodes, which are discussed in the following section.

4.3.4 Detect boundary Cells and nodes

First, it is crucial to define the boundary cells and nodes. The boundary cell is the cell that contains both fluid and solid nodes. On the other hand, the boundary node is the solid node that has neighboring fluid nodes. Therefore, the boundary cells are detected by first detecting the fluid and solid cells, which contain all the nodes as fluid and solid nodes, respectively. The fluid and solid cells can be detected using the following two conditional arrays.

$$CA_S(i_{cell}) = [(S_N(Node(i_{cell}, P_1)) = 1) \cdot (S_N(Node(i_{cell}, P_2)) = 1) \cdot (S_N(Node(i_{cell}, P_3)) = 1) \cdot (S_N(Node(i_{cell}, P_4)) = 1)] \quad (4.57)$$

$$CA_F(i_{cell}) = [(F_N(Node(i_{cell}, P_1)) = 1) \cdot (F_N(Node(i_{cell}, P_2)) = 1) \cdot (F_N(Node(i_{cell}, P_3)) = 1) \cdot (F_N(Node(i_{cell}, P_4)) = 1)] \quad (4.58)$$

Then, the boundary cells $CA_B(i_{cell})$ can be defined as the remaining cells that are not detected by solid and fluid conditional arrays; consequently, their corresponding indices iB_{cell} as

$$CA_B(i_{cell}) = 1 - CA_S(i_{cell}) - CA_F(i_{cell}) \quad (4.59)$$

$$iB_{cell} = i_{cell} \quad \text{if } CA_B(i_{cell}) = 1 \quad (4.60)$$

After that, the boundary cells are tested to detect the boundary nodes. There are 12 possible combinations where a solid node(s) has a neighboring fluid node(s) inside a boundary cell. These 12 cases are shown in Figure 4.9. Finally, the twelve cases are implemented, and the index where a solid node is found for all the cases is set as a boundary node. For instance, in case #1, the solid node is the point P_1 , consequently, the node boundary array is set to one corresponding to the P_1 index as $B_N(Node(iB_{cell}, P_1)) = 1$. Consequently, the boundary node indices iB_{Node} can be detected as

$$iB_{Node} = i_{Node} \quad \text{if } B_N(i_{Node}) = 1 \quad (4.61)$$

After detecting the boundary nodes, the solid array is redefined to remove any boundary nodes from it to distinguish between a fully solid node, where no calculations are applied, and a boundary node as:

$$S_N(i_{Node}) = S_N(i_{Node}) - B_N(i_{Node}) \quad (4.62)$$

The boundary detection algorithm is tested using NACA 2412 airfoil. The fluid, solid, and boundary nodes are correctly detected overall in the domain, as shown in Figure 4.10(a). Moreover, a close-up view of the trailing edge is shown in Figure 4.10(b), which clearly shows the correct detection of the boundary nodes, even if the node is very close to the physical wall.

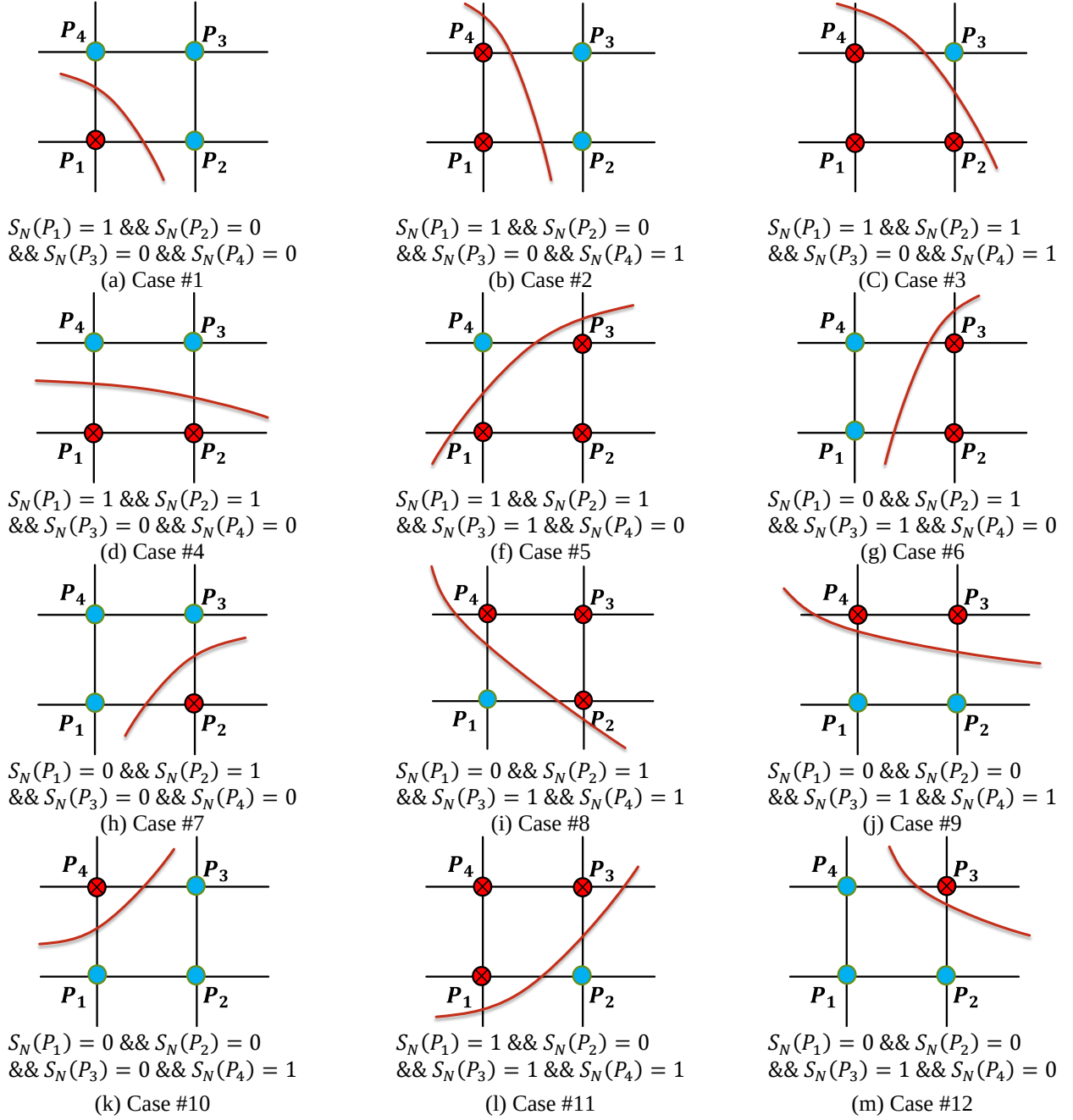


Figure 4.9. Schematic showing the twelve possible combinations of the solid and fluid nodes inside a boundary cell. The blue circles are the fluid nodes, the red line is the physical boundary, and the red crossed circles are the solid nodes defined as boundary nodes.

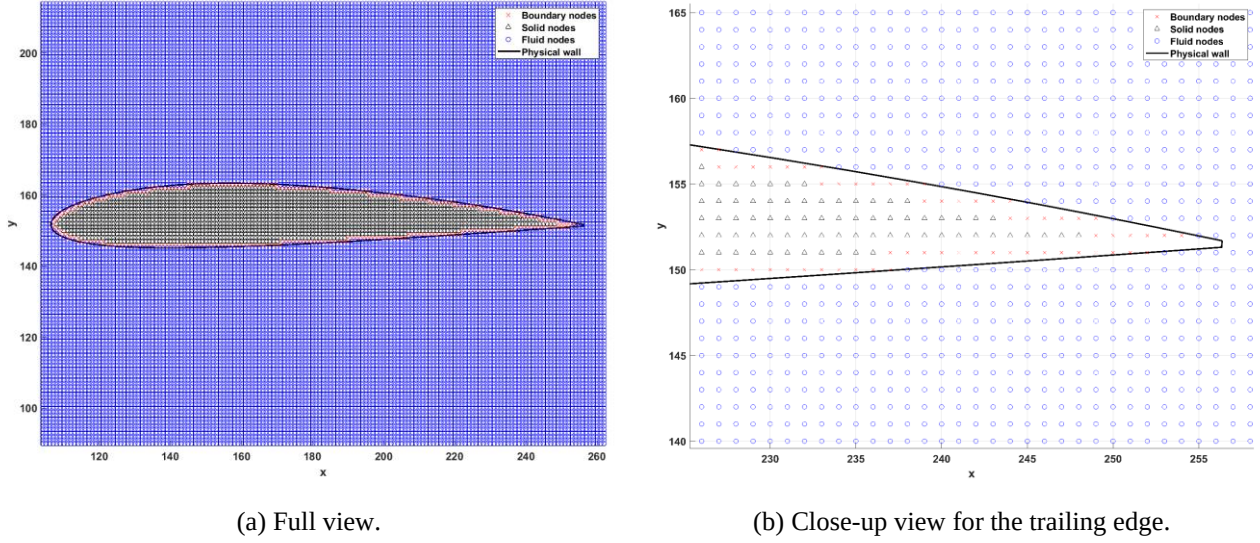


Figure 4.10. A visualization of solid, boundary, and fluid nodes for NACA 2412 airfoil.

4.3.5 Displacement calculation

The displacement function Δ is required to calculate the extrapolation values shown in equations (4.21) and (4.24). This displacement function is calculated from equation (4.22) for all boundary nodes iB_{Node} in the direction where fluid nodes exist.

The subscript i is the direction of the displacement, similar to the particle distribution function direction. The position vector $\mathbf{x}_{wi}$ is the location of the physical wall intersection with the cell in the i direction, the boundary node location $\mathbf{x}_B$, and the neighboring fluid node in the i direction is $\mathbf{x}_{fi}$. This equation could be written using the study data structure:

$$\Delta(iB_{Node}, i) = \frac{\|\mathbf{x}_w(iB_{Node}, i) - \mathbf{x}(iB_{Node})\|}{\|\mathbf{x}(iB_{Node}) - \mathbf{x}(Sur(iB_{Node}, i))\|} \quad (4.63)$$

Figure 4.11 shows the directions for the displacement calculations and their corresponding intersection points for case#1 and case#2 as examples. For case #1, the boundary node is P_1 and the displacement function is calculated in directions 1, 2, and 5 as they point to fluid nodes. For example, in direction 5, the $\mathbf{x}_w(iB_{Node}, i)$ is the location of the plus-empty circle $\oplus$, which indicates the intersection of the physical wall with the diagonal direction, and the $\mathbf{x}(Sur(iB_{Node}, i))$ is the location of P_3 . Similarly, in the remaining two directions. On the other hand, case#2 has two boundary points P_1 and P_4 , the displacement function is calculated in directions 1 and 5 for P_1 and directions 1 and 8 for P_4 . There is no displacement calculation between

the boundary nodes as the displacement is only calculated between a fluid and boundary nodes. The other cases are implemented similarly.

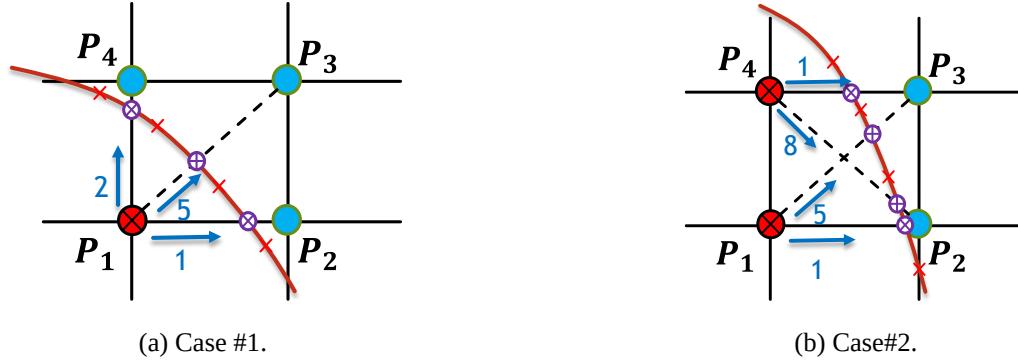


Figure 4.11. Schematic showing the intersection points required for the displacement function calculations. The blue circles are the fluid nodes, the red line is the physical boundary, the red crossed circles are the boundary nodes, the dashed line is the cell diagonal, the cross-empty circles $\otimes$ are the intersection of the cell with the physical wall, the plus-empty circle is the intersection of the cell's diagonal with the physical wall $\oplus$, and the red cross markers are the arbitrary data for the physical wall.

Figure 4.12. shows an example of a boundary node iB_{Node} surrounded by four fluid nodes in the directions 1, 5, 2, and 6. Applying case#1 on $Cell_1$ covers the directions 1, 5, and 2, while direction 6 is not covered. However, direction 6 is covered when applying case#4 in $Cell_2$. Consequently, The screening over the boundary cells covers all the possibilities of calculating the displacement function in all directions.

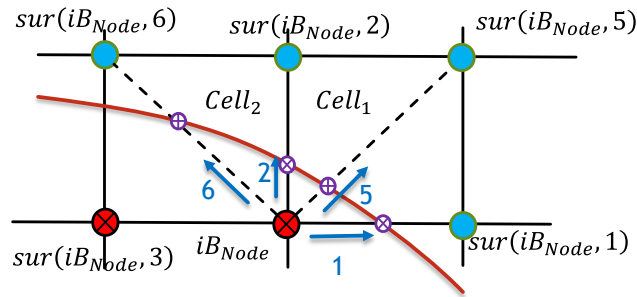


Figure 4.12. Schematic showing the intersection points required for the displacement function calculations.

The intersection points locations are calculated using the two lines' intersection using four points shown in Figure 4.13. , the corresponding equations are defined as

$$x_w = \frac{(x_1y_2 - x_2y_1)(x_3 - x_4) - (x_1 - x_2)(x_3y_4 - x_4y_3)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)} \quad (4.64)$$

$$y_w = \frac{(x_1y_2 - x_2y_1)(y_3 - y_4) - (y_1 - y_2)(x_3y_4 - x_4y_3)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)} \quad (4.65)$$

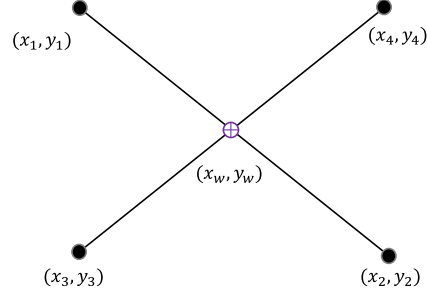


Figure 4.13. Intersection point schematic using two lines intersection method.

The values of the four points' coordinates differ depending on the cases. For example, in case#1, direction 5, (x_1, y_1) and (x_2, y_2) are the successive physical wall arbitrary points, while (x_3, y_3) is the boundary node and (x_4, y_4) is the fluid node. In another case, such as case#4, direction 6, (x_1, y_1) is the fluid node, (x_2, y_2) is the boundary node, and (x_3, y_3) and (x_4, y_4) are successive physical wall arbitrary points.

4.3.6 Mesh generation algorithm Procedures

Figure 4.14. shows the flow chart of the mesh generation algorithm, which could be mentioned in the following procedures:

- Define the nodes and surrounding arrays mentioned in section 4.3.1.
- Detect the solid nodes using the winding number algorithm. Then, define the solid and fluid nodes' arrays mentioned in section 4.3.4.
- Detect the boundary cells array and define the boundary cell array, then loop over these boundary cells to detect the boundary nodes using the 12 cases discussed in section 4.3.4.
- Calculate the intersection points using the intersection function, then calculate the displacement function illustrated in section 4.3.5.
- Loop over all the boundary cells, then exports the nodes array, surrounding array, fluid, solid, boundary node arrays, and the displacement function to the main program.

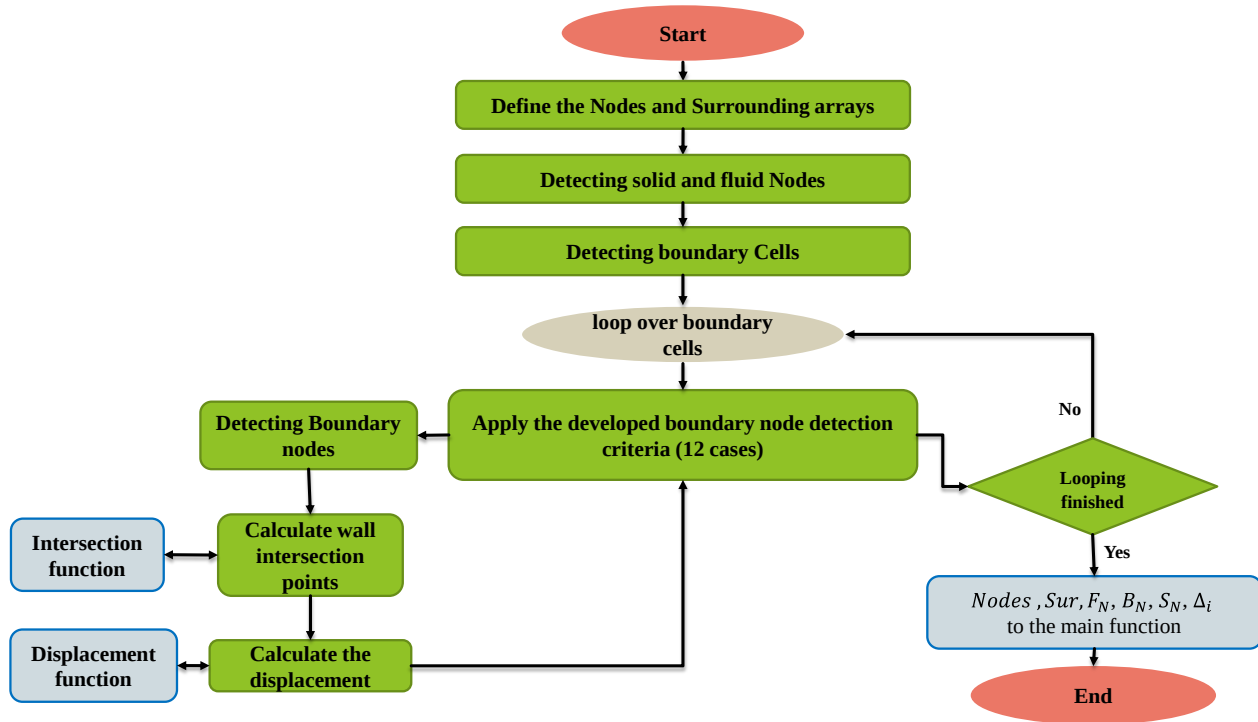


Figure 4.14. The flow chart of the mesh generation algorithm.

4.4 LBM with extrapolation technique implementation

Figure 4.15. is the flow chart for the LBM with the extrapolation technique illustrating the implementation procedures as follows:

- The domain variables, such as the Reynolds number, grid size, and Mach number, are defined.
- The mesh generation algorithm procedures are implemented, as illustrated in section 4.3.6.
- The displacement function conditional arrays are calculated from equations (4.26) and (4.27).
- Apply the initial conditions on the velocity and density, then construct the initial PDFs using its equilibrium part as in equation (4.4). After that, apply the inlet/outlet boundary conditions, such as Zou and He boundary conditions as illustrated in section 4.2.1, then start the time loop.
- Calculate the equilibrium particle distribution function on the fluid nodes using equation (4.4), then calculate the nonequilibrium part using equation (4.18).

- f) Extrapolate the macroscopic on the boundary nodes from the fluid nodes using equations (4.19) and (4.28).
- g) Reconstruct the equilibrium particle distribution function on the boundary nodes using the extrapolated macroscopic variables obtained in the previous step.
- h) Extrapolate the non-equilibrium particle distribution function on the boundary nodes from the surrounding fluid node using equation (4.29).
- i) Reconstruct the pre-collision particle distribution function on the boundary node using equation (4.18).
- j) Apply the collision and streaming on the fluid and boundary nodes using equations (4.2) and (4.3).
- k) Apply the boundary conditions, such as the halfway bounce-back discussed in 4.2.3 or the slip boundary condition illustrated in 4.2.2, and the inlet/outlet boundary conditions, such as Zou and He boundary conditions.
- l) Update the macroscopic variables using equations (4.5) and (4.6) on the fluid nodes.
- m) Repeat steps from e) to l) until the simulation ends.

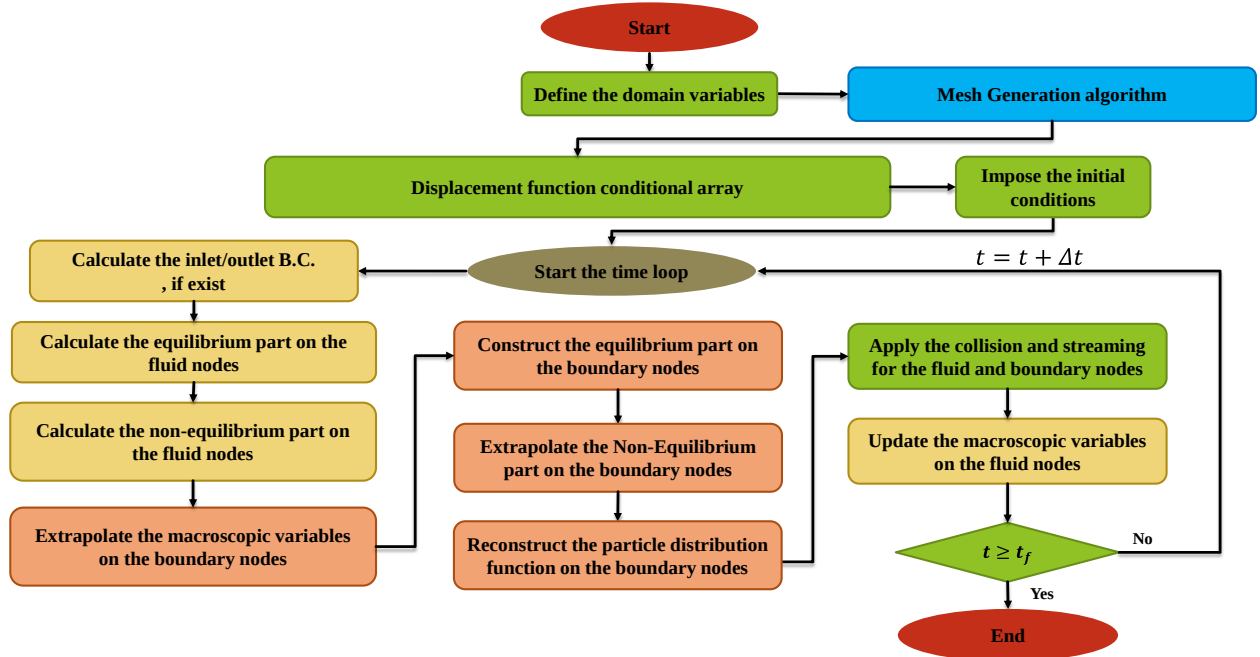


Figure 4.15. The flow chart of the LBM with the extrapolation technique.

4.5 Results and discussion

This section illustrates the verification and code-to-code verification of implementing LBM with the extrapolation method through four test cases. First, in section 4.5.1, the Poiseuille flow is implemented to verify the standard LBM by retrieving the analytical solution velocity profile and second-order accuracy in space. Then, section 4.5.2 illustrates the Taylor-Couette flow implementations, which verify the LBM with the extrapolation method. After that, in section 4.5.3, the uniform flow around a stationary circular cylinder is introduced to validate LBM with the extrapolation technique without imposing velocity on the boundaries. Finally, section 4.5.4 solves the flow around a rotating cylinder to validate the extrapolation LBM with velocity imposed on the boundaries.

4.5.1 Poiseuille flow

In this section, the Poiseuille flow is implemented to verify the standard LBM by calculating the spatial order of accuracy and comparing the velocity profile with the analytical solution. In addition, two boundary conditions combinations are implemented, shown in Table 4.1, where a refinement study is performed for each test case.

Table 4.1 Poiseuille flow Test cases for different boundary conditions.

Case	Inlet	Outlet	Upper	Lower
a	Velocity (Zou and He)	Pressure (Zou and He)	Halfway bounce-back	Halfway bounce-back
b	Periodic	Periodic	Halfway bounce-back	Halfway bounce-back

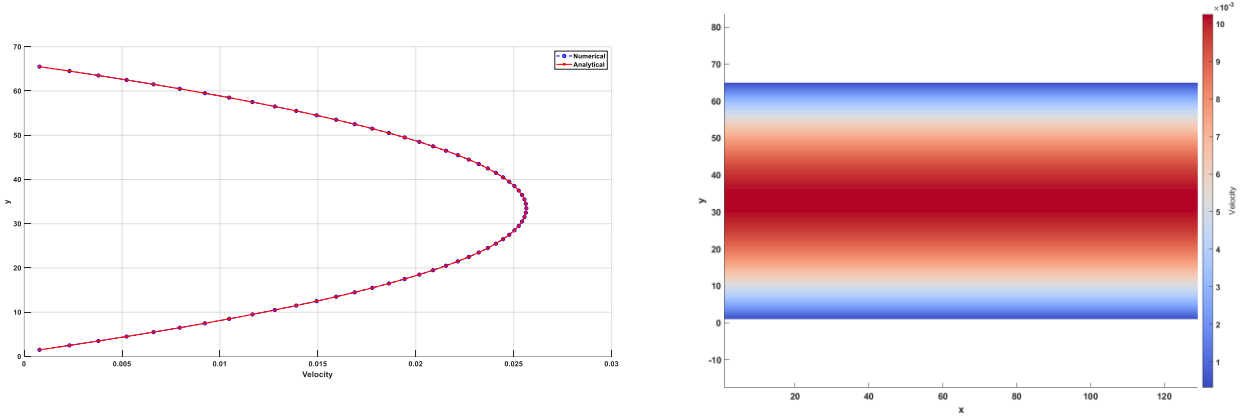
The idea of implementing different boundary conditions is to compare the L2 velocity error norm e_2^u besides investigating the order of accuracy for each boundary condition. The e_2^u is calculated as:

$$e_2^u = \sqrt{\frac{\sum_{k=0}^{N-1} (u(x_k) - u_{ana}(x_k))^2}{N}} \quad (4.66)$$

Where k is the node index with its corresponding length N . The analytical solution u_{ana} for Poiseuille flow, which is also imposed on the inlet boundary regarding case (a), is defined as:

$$u_{ana}(x_k) = u_{max} \left(\frac{4y_k}{L} - \frac{4y_k^2}{L^2} \right) \quad (4.67)$$

A series of simulations are performed with grid sizes 32×16 , 64×32 , 128×64 , and 256×128 , with a fixed value of the relaxation time $\tau = 1$, and Reynolds number $Re = 10$ for both cases. For example, the velocity profile for case (a) is presented in Figure 28, showing that the numerical solution matches the analytical one.



(a) The velocity profile at half of the length $N_x/2$.

(b) The velocity field for numerical solution. The analytical solution is similar.

Figure 4.16. Numerical and analytical velocity profile and field for grid size 128×64 with halfway bounce-back. Other case has similar profiles.

Table 4.2 shows e_2^u and the order of accuracy, where the two cases retrieve a third order of accuracy in space. Moreover, the periodic boundary condition, case (b), has a lower L2 error norm at the same grid size than the Zou and He boundary conditions.

Table 4.2 Poiseuille flow e_2^u and order of accuracy for different boundary conditions. Cases (a) and (b) are described in Table 4.1

Case	Refinements	Grid size	e_2^u	Order of accuracy
a	1	32×16	1.9544857885812E-03	
	2	64×32	2.5814711216126E-04	2.920523
	3	128×64	3.3510186595896E-05	2.945521
	4	256×128	4.2799664210998E-06	2.968928
b	1	32×16	1.1836605707536E-04	
	2	64×32	1.5821006471549E-05	2.903342
	3	128×64	2.0465992124738E-06	2.950540
	4	256×128	2.5871574218403E-07	2.983788

Furthermore, another refinement study is implemented for case (a) with relaxation time $\tau = 0.7$, and Reynolds number $Re = 10$ to investigate the effect on the order of accuracy and e_2^u , shown in Table 4.3. It was found that the order of accuracy is also a third-order accuracy. Moreover, the L2 error norm is less than the same test case with $\tau = 1.0$.

Table 4.3 Poiseuille flow e_2^u and order of accuracy for case (a) with $\tau = 0.7$.

Case	Refinements	Grid size	e_2^u	Order of accuracy
a	1	32×16	1.4340151508435E-04	
	2	64×32	1.9487318422671E-05	2.87945
	3	128×64	2.5468077785647E-06	2.93577
	4	256×128	3.2633770868409E-07	2.96425

The order of accuracy of the two test cases has a third order of accuracy for different relaxation times. These results add a new claim besides the one stated by Kruger et al. [28], as the third-order accuracy can be obtained in the physical domain **if and only if** $\tau = 0.93 \Delta t$. However, using the lattice system of units changes this claim as for any value of τ in the lattice system of units retrieves a third order of accuracy in space, which is discussed in the following paragraphs.

The third-order spatial accuracy will be achieved if the velocity on the wall $u_x|_{y_w}$ must satisfy the second-order Taylor expansion [28]. The Taylor expansion of the wall velocity on the upper boundary is

$$u_x|_{y_w} = u_x|_{y_b} + \frac{\Delta x}{2} \partial_y u_x|_{y_b} + \frac{1}{2} \left(\frac{\Delta x}{2} \right)^2 \partial_y \partial_y u_x|_{y_b} + O(\Delta x)^3 \quad (4.68)$$

Where y_b is the location of the boundary nodes. Using the lattice system of units substitutes the displacement as unity $\Delta x = 1$. Moreover, the velocity derivatives in the y-direction are implemented through equation (4.67), leading to

$$u_x|_{y_w} = \frac{4 u_{max}}{L^2} \left[(y L - y^2) + \frac{1}{2} (L - 2y) + \left(\frac{-2}{8} \right) \right] \Big|_{y=y_b} \quad (4.69)$$

Now, by substituting the boundary location y as the midway distance $y_b = (L - 0.5)$, we can find that:

$$u_x|_{y_w} = \frac{4 u_{max}}{L^2} \left[((L - 0.5)L - (L - 0.5)^2) + \frac{1}{2} (L - 2(L - 0.5)) - 0.25 \right]$$

$$u_x|_{y_w} = \frac{4 u_{max}}{L^2} [(L^2 - 0.5L - L^2 + L - 0.25) + (0.5L - L + 0.5) - 0.25] = 0$$

Thus, the velocity on the physical wall retrieves its exact solution from the second-order Taylor expansion with $\Delta x = 1$. Consequently, applying $\Delta x = 1$ is a new exceptional case in the halfway bounce-back, besides the case mentioned in Kruger et al. for the physical system of units, retrieving a third-order accuracy despite the value of relaxation time.

4.5.2 Taylor-Couette flow

This section is presented to verify the standard LBM with the curved boundary extrapolation method. The Taylor-Couette flow is the flow presented between two concentric cylinders. The current test case imposed a rotational speed on the inner cylinder boundary and zero velocity on the outer cylinder. The domain is selected as a square with an outer cylinder radius $R_{out} = 0.45 N_x$. The inner cylinder radius R_{in} is calculated using the radii ratio $\beta = R_{in}/R_{out}$. The relaxation time is selected as $\tau = 1.0$, and the Reynolds number $Re = 10$. Moreover, the flow is only solved on the nodes between those two cylinders. The analytical solution [89] of the corresponding test case can be presented as

$$u_{ana}(x, y) = u_{max} \left(\frac{\beta}{1 - \beta^2} \right) \left(\frac{R_{out}}{r(x, y)} - \frac{r(x, y)}{R_{out}} \right) \quad (4.70)$$

Moreover, the simulation is terminated once it reaches a stopping criteria defined as follows:

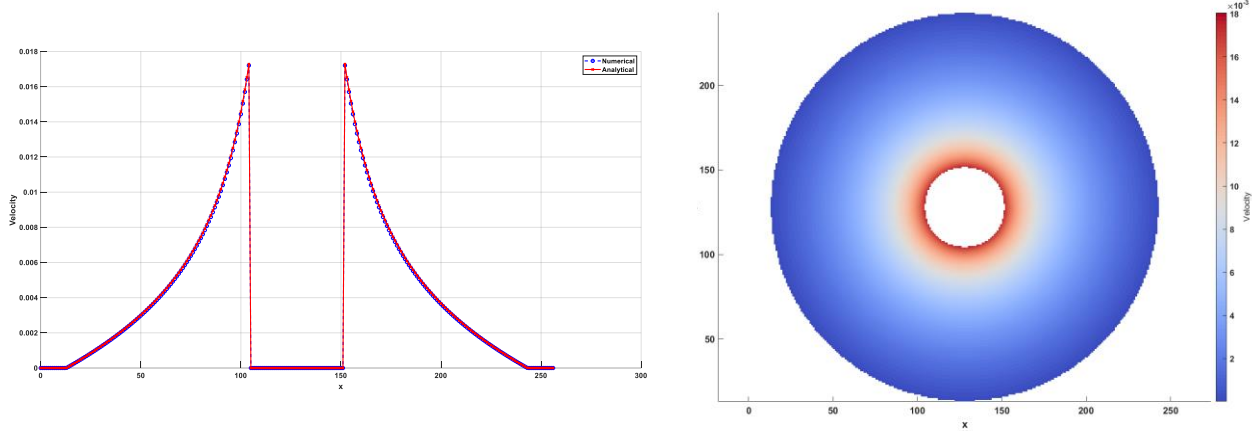
$$\frac{\sum_{x_f} \|u(\mathbf{x}_f, t + \Delta t) - u(\mathbf{x}_f, t)\|}{\sum_{x_f} \|u(\mathbf{x}_f, t)\|} < 10^{-10} \quad (4.71)$$

The velocity field and the velocity profile for the numerical solution are presented in Figure 4.17, showing the matching between the analytical and numerical velocity profiles.

Furthermore, the L2 relative velocity error $\hat{e}_2^u$ is calculated for retrieving the order of accuracy, which is calculated as

$$e_2^u = \sqrt{\frac{\sum (u(\mathbf{x}_f) - u_{ana}(\mathbf{x}_f))^2}{N_f}} \quad (4.72)$$

Where $\mathbf{x}_f$ is the fluid node spatial vector and N_f is the total number of fluid nodes. Moreover, Figure 4.18 shows the second order of accuracy retrieved by e_2^u , which verifies the extrapolation LBM.



(a) The velocity profile at half of the height $N_y/2$.

(b) The velocity field for numerical solution. The analytical solution is similar.

Figure 4.17. Taylor-Couette flow numerical and analytical velocity profile and field for grid size 256×256 . Other case has similar profiles.

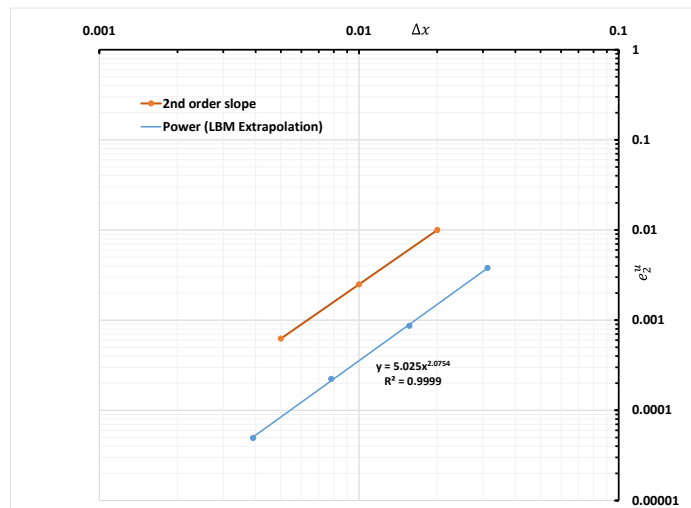


Figure 4.18. Taylor-Couette order of accuracy.

4.5.3 Uniform flow around a circular cylinder

The uniform flow around the rotating cylinder is implemented to validate the LBM with the extrapolation method. The inlet and outlet boundary conditions are the Zou and He velocity input and pressure output boundary conditions, while the far field of the lower and upper surfaces is straight velocity implemented according to the Inamuro boundary condition. Moreover, the circular cylinder is implemented using the curved boundary extrapolation treatment with zero velocity imposed on the boundary.

The simulation parameters are the grid size of 1800×850 nodes, Reynolds number $Re = 100$, reference length $L = D = N_y/16$, the relaxation time $\tau = 0.65$, the kinematic viscosity $\nu = \frac{1}{3}(\tau - 0.5)$, the freestream velocity $u_\infty = \frac{Re \nu}{L} = 0.09411$. The simulation starts with zero velocity and unity density in the overall domain. Figure 4.19 shows the velocity field around the circular cylinder, showing the propagation of the vortex street.

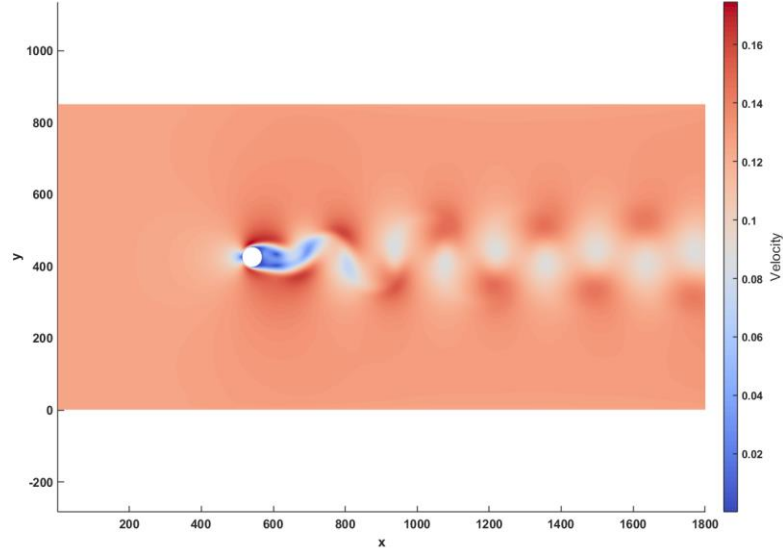


Figure 4.19. Velocity field for flow around rotating cylinder at $Re=100$.

The total forces acting on the cylinder boundary $\mathbf{F}_B$ are calculated according to the momentum exchange theory [32] as

$$\mathbf{F}_B = \sum_{k \in B_n} \sum_{i=1}^8 \xi_i (f_i^*(\mathbf{X}_k, t) + f_i^*(\mathbf{X}_k + \xi_i \delta_t, t)) \cdot \delta_k \quad (4.73)$$

Where B_n is the set of the boundary nodes indices and δ_k equals one if the node is fluid and zero otherwise. The boundary nodes are the nodes that have both fluid and solid nodes in their neighbors. The post-collision particle distribution function at the boundary nodes $f_i^*(\mathbf{X}_b, t)$, and $f_i^*(\mathbf{X}_b + \xi_i \delta_t, t)$ is the post-collision particle distribution function of the neighboring nodes in the opposite direction. Moreover, the lift and drag coefficients in zero angle of attack are calculated from $\mathbf{F}_B = F_x \hat{i} + F_y \hat{j}$ as

$$C_L = \frac{2 F_y}{\rho u_\infty^2 D} \quad (4.74)$$

$$C_D = \frac{2 F_x}{\rho u_\infty^2 D} \quad (4.75)$$

Table 4.4 shows the lift coefficient amplitude C'_L , mean drag coefficient $\bar{C}_D$ and Strouhal number $St = f L/u_\infty$, where f is the vortex shedding frequency. The results show good agreement for the metrics with the published data, solving a similar external flow. These results validate the LBM with extrapolation technique for unsteady flow.

Table 4.4 The lift and drag coefficients and Strouhal number for $Re=100$.

Study	C'_L	$\bar{C}_D$	St
Present	0.34	1.399	0.1679
Lima et al. [90]	0.29	1.39	0.164
Tseng and Ferziger [91]	0.29	1.42	0.164
Lai and Peskin [92]	0.3299	1.447	0.165
Berthelsen and Faltinsen [93]	0.34	1.38	0.169

4.5.4 Uniform flow around rotating circular cylinder

In this section, two simulations are performed. The first simulation parameters are the grid size of 1200×600 , rotation ratio $a = 0.5$, Reynolds number $Re = 20$, reference length $L = D = N_y/16$, the relaxation time $\tau = 0.7$, the kinematic viscosity $\nu = \frac{1}{3}(\tau - 0.5)$, the freestream velocity $u_\infty = \frac{Re \nu}{L} = 0.0356$, angular speed $\omega = u_\infty a/r = 0.0009493$. The simulations start with zero velocity and unity density in the overall domain, with the rotation starting from the first time step. The second simulation parameters are the grid size of 1200×600 , rotation ratio $a = 1.0$, Reynolds number $Re = 100$, reference length $L = D = N_y/16$, the relaxation time $\tau = 0.6$, the kinematic viscosity $\nu = \frac{1}{3}(\tau - 0.5)$, the freestream velocity $u_\infty = \frac{Re \nu}{L} = 0.0889$, angular speed $\omega = u_\infty a/r = 0.0047407$. The simulations start with zero velocity and unity density in the overall domain, with the rotation starting from the first time step.

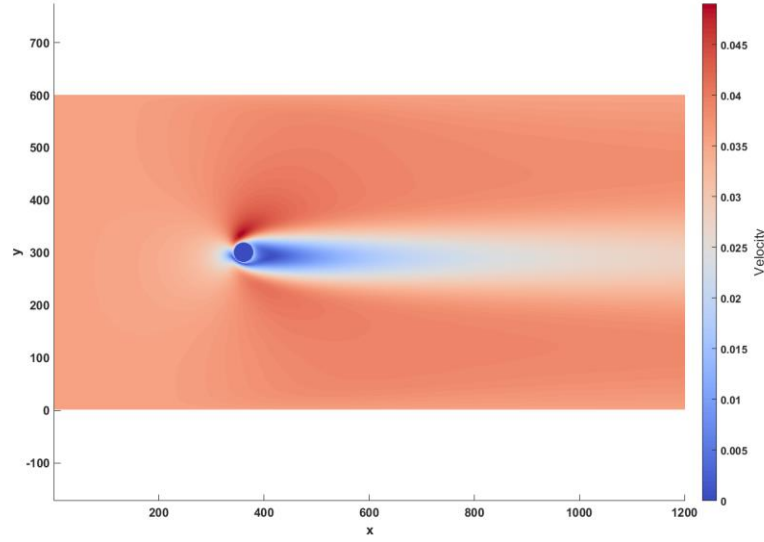


Figure 4.20. Velocity field for flow around rotating cylinder at $Re=20$, $a=0.5$.

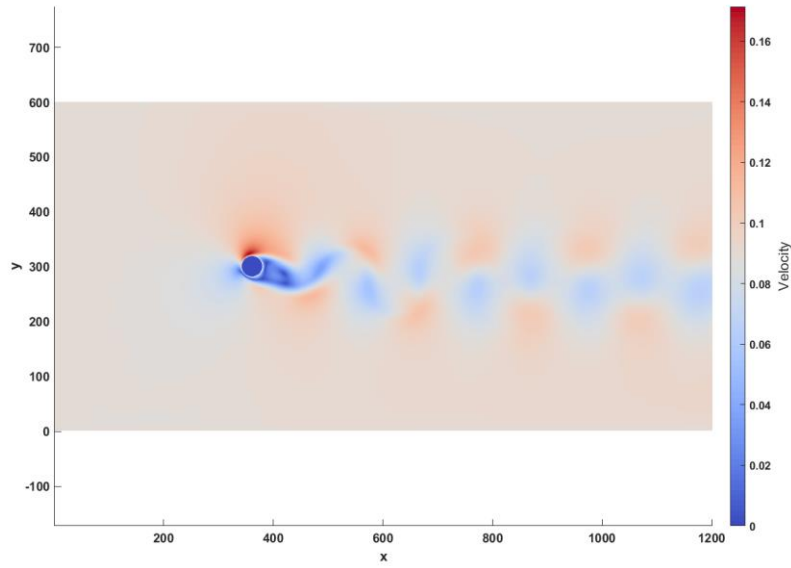


Figure 4.21. Velocity field for flow around rotating cylinder at $Re=100$ and $a=1.0$.

Figure 4.20 and Figure 4.21 qualitatively validate the implementation of LBM with Guo's extrapolation method by presenting the velocity field around the rotating cylinder for Reynolds numbers 20 and 100. For $Re = 20$, the velocity field flow deformation is presented due to the rotation of the cylinder by enforcing a velocity profile on the cylinder boundary. Moreover, the low Reynolds number develops the flow as a steady solution as expected. In the case of $Re = 60$, the velocity profile is developed into an unsteady flow field with a vortex street as the Reynolds

number is high enough to develop this phenomenon, and the rotational ratio did not reach its steady state value a_s (more discussion about the steady state rotational ratio is in Chapter 6).

Moreover, a code-to-code verification is implemented by calculating the mean lift coefficient $\bar{C}_L$, the amplitude lift coefficient C'_L , the drag coefficient, $\bar{C}_D$ and Strouhal number St . The mean lift coefficient $\bar{C}_L$ is calculated using the peak values of the lift coefficient, maximum C_L^{max} and minimum C_L^{min} values of the lift coefficient as $\bar{C}_L = \frac{C_L^{max} + C_L^{min}}{2}$, while the amplitude lift coefficient C'_L is calculated as $C'_L = \frac{C_L^{max} - C_L^{min}}{2}$.

Table 4.5 The lift and drag coefficients for rotation ratio = 0.5 and Re=20.

Study	$\bar{C}_L$	$\bar{C}_D$
Present	1.4224	2.177
Kang et al. [94]	1.363	–
Badr et al. [95]	1.4057	2.172

Table 4.6 The lift and drag coefficients and Strouhal number for rotation ratio = 1.0 and Re=100.

Study	$\bar{C}_L$	C'_L	$\bar{C}_D$	St
Present	2.5187	0.4053	1.2045	0.167
Kang et al. [94]	2.48	0.38	1.12	0.165

Table 4.5 shows the lift and drag coefficients for the steady solution where the Reynolds number equals 20, and Table 4.6 shows the lift and drag coefficients and Strouhal number for the unsteady solution where the Reynolds number equals 100, and the results were found to be matched for all metrics with the available references [94, 95].

In summary, the preceding test cases verify and validate the standard BGK LBM in combination with the extrapolation technique and various boundary conditions for treating curved boundaries. The Poiseuille flow test case verifies the LBM's effectiveness when employing the halfway bounceback method and demonstrates that using LB system units can achieve third-order spatial accuracy in such scenarios. Furthermore, the Taylor-Couette flow case verifies the LBM with the non-equilibrium extrapolation approach for handling curved boundaries, retrieving a second-order accuracy and a matched numerical velocity profile with the analytical one. Lastly, the flow simulations around stationary and rotating cylinders are employed to validate the

implementation of LBM with curved boundary treatments, yielding results consistent with previously published data.

CHAPTER 5 **ARTICLE 1: Impact of interpolation method selection on the accuracy of the lattice Boltzmann-overset method**

This chapter is a submitted manuscript that discusses the implementation of the LB-overset method and the impact of the interpolation methods on it.

Abdallah ElSherbiny, Nicolas Pellerin, and Sébastien Leclaire, “Impact of interpolation method selection on the accuracy of the lattice Boltzmann-overset method,” submitted to *Computers & Mathematics with Applications* on Jun 9, 2023.

5.1 Abstract

The integrated lattice Boltzmann-overset method is a method that solves the flow around moving boundaries. This method's domain is divided into two grids with their respective coordinate systems: the fixed grid and the moving grid, where the solid body is tied to the latter. Both grids share an overlapping region and communicate through two interfaces acting as boundary conditions, which are implemented by interpolating the macroscopic quantities between the grids. The proposed study compares six interpolators on the 2D LB-overset method based on the recursive regularized Bhatnagar–Gross–Krook collision model. These interpolators are the nearest neighbor, inverse distance weighting, radial basis function, bilinear, bicubic convolution algorithm, and full bicubic interpolators. These interpolators were selected for testing interpolation methods applicable to uniform and non-uniform grids besides testing the different polynomial orders. The selection criteria are based on the solution's stability and accuracy, which could be illustrated by the velocities discrepancy and the pressure noise resulting from the moving grid motion. The test cases used are based on flow steadiness, such as Couette flow and high Reynolds advection vortex in uniform flow. The study revealed that the inverse distance weighting, bilinear, bicubic convolution, and bicubic interpolators behave similarly for steady flows. However, the bicubic convolution and bicubic interpolators have the most accurate and stable solutions for the unsteady flow.

5.2 Introduction

The overset method is widely applied in computational fluid dynamics, especially with the Navier Stokes solvers [75-77]. It was first used in the 1980s in studying an airfoil by connecting a body-fixed O-type grid with a cartesian grid called the CHIMERA grid [78]. Later, many studies

developed the overset method to solve the moving boundary problem with the hole-cutting approach [79, 80]. This approach has two overlapped grids: the refined grid conforming to the body wall and a coarse fixed grid. Those two grids communicate through interpolation on donor and receptor nodes on a fringe zone. Moreover, all the fixed grid nodes in the overlapped region are deactivated. So, the hole-cutting process refers to detecting the donor and receptor nodes and deactivating the fixed grid redundant nodes. However, this process must be evaluated with the refined grid motion at each time step, leading to a high processing time. Consequently, the moving overset grids method is developed where some interpolation or extrapolation techniques apply the connection between the moving and fixed grids without altering the grid structure [81].

The moving grid overset method is first introduced with the LBM solver as a sliding mesh interface method [54, 82, 83] because the domain is separated into two grids sharing only one axisymmetric interface ring suitable for rotating boundaries [54]. Then, the LB-overset method is introduced by defining two communication boundaries, fixed-to-moving and moving-to-fixed, which reduces the numerical noise generated from the relative motion [55, 56]. The space between these two boundaries is called the buffer zone, which should be at least twice the grid spacing to allow the interpolation to be evaluated [55]. Moreover, applying the two boundaries of communication reduces the numerical noise due to the relative motion of the moving mesh.

The LB-overset method could be clearly defined as a second-order accuracy method [55, 56, 61] with a domain consisting of two grids with their respective coordinate systems; these grids are the fixed grid and the moving grid to which the solid body is attached. Both grids share an overlapping region and communicate through two interfaces acting as boundary conditions [56]. This communication on the borders is performed by interpolating the macroscopic variables between the two grids. Moreover, most previous studies use the bilinear interpolator with nine surrounding nodes [55, 56]. These nine nodes are selected from the target node current, west, south, and southwest cells, which neglect the other surrounding cells.

Furthermore, another study compares the bilinear interpolator and the gradient-based quadratic Hermite interpolation [61]. These interpolators use only the four surrounding points and have a second order of accuracy [61]. Moreover, the gradient-based interpolator is better than the bilinear interpolator as it suppresses the noise resulting from the border interpolations. However, the gradient-based interpolation is based on the strain tensor calculation, which may need further

information from the surrounding cells in the case of single relaxation time (SRT) LBM, where the strain tensor is not calculated. Thus, the gradient-based interpolator may be more suitable with the multi-relaxation time (MRT) LBM as the strain tensor must be calculated, where it was first introduced [84].

The present study analyzes the impact of the interpolator selection on the lattice Boltzmann-overset method and finds the most suitable one. The nearest neighbor, inverse distance weighting (IDW), radial basis function (RBF), bilinear, bicubic-convolution, and bicubic interpolators are introduced, covering a variety of interpolator properties. For instance, mesh-free interpolators are presented in IDW and RBF interpolators. Moreover, the interpolator node number is also covered based on four and sixteen nodes, such as the bilinear and bicubic interpolators. In addition, the IDW and RBF are implemented twice with four (IDW4, RBF4) and sixteen (IDW16, RBF16) nodes as they can handle any number of nodes, allowing us to compare the effect of nodes in each interpolator.

The lattice Boltzmann method using the recursive regularized BGK model is first explained briefly in section 2, and for more details, the reader could return to the references [28, 60, 62]. Later, the overset method based on Yoo et al. study [61] is discussed in section 3, showing the borders' communication. After that, the computational procedures are described in section 4. Next, the interpolators are illustrated in section 5, representing the stencil detection algorithm and the interpolators' corresponding equations besides the procedures. Then, the results are discussed in section 5 by evaluating five test cases. These test cases are uniform flow, Poiseuille flow, Taylor-Couette flow, high-Re vortex advection in uniform flow, and uniform flow around a rotating cylinder, where the flow complexity gradually increases. Finally, the conclusion is mentioned in section 6, and an appendix is attached at the end of the paper containing the equations related to the recursive regularized LBM and the table of interpolators verification.

5.3 The Lattice Boltzmann method using the recursive regularized BGK model

The lattice Boltzmann method solves the fluid motion based on the kinetic theory. It discretized the lattice Boltzmann equation (LBE) into time and space with a discretized velocity space ξ_i using Gauss-Hermite quadrature discretization to evaluate the particle distribution function $f_i(\mathbf{x}, t)$ in a specific location, time, and velocity direction. The LBM core relies on two main steps: collision and streaming. The collision step represents colliding the incoming particle distribution function

and then constructing outgoing particle distribution functions depending on its discretized velocity space direction. The streaming step represents the propagation of the outgoing particle distribution functions to the surrounding lattices. The LBE with no external force can be written as

$$f_i(\mathbf{x} + \boldsymbol{\xi}_i \Delta t, t + \Delta t) - f_i(\mathbf{x}, t) = \Omega_i(\mathbf{x}, t) \Delta t \quad (5.1)$$

Where $\Omega_i(\mathbf{x}, t)$ is the collision operator, and Δt is the time step. In its original form, this collision model is a complex double integral over the velocity spaces, which considers all the possible outcomes of two-particle collisions for intermolecular forces [28]. This collision operator can be linearized based on the Bhatnagar–Gross–Krook (BGK) model [96], which defines the collision operator with a single relaxation of the distribution function to its local equilibrium distribution $f_i^{eq}(\mathbf{x}, t)$. The LBE can be written as

$$f_i(\mathbf{x} + \boldsymbol{\xi}_i \Delta t, t + \Delta t) - f_i(\mathbf{x}, t) = -\frac{\Delta t}{\tau} \left(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t) \right) \quad (5.2)$$

Where τ is the relaxation time, and the equilibrium distribution can be defined as

$$f_i^{eq}(\mathbf{x}, t) = w_i \rho \left(1 + \frac{\boldsymbol{\xi}_i \cdot \mathbf{u}}{c_s^2} + \frac{1}{2 c_s^4} \mathcal{H}_i : \mathbf{u} \mathbf{u} \right) \quad (5.3)$$

Where $\mathcal{H}_{i\alpha\beta} = \xi_{i\alpha} \xi_{i\beta} - c_s^2 \delta_{\alpha\beta}$, the lattice speed of sound is c_s , the lattice weights is w_i , and $\delta_{\alpha\beta}$ is the Kronecker delta. The density ρ and velocity $\mathbf{u}$ can be calculated from the zeroth, and the first moments of the particle distribution function for the D2Q9 model as

$$\rho = \sum_{i=0}^8 f_i \quad (5.4)$$

$$\rho \mathbf{u} = \sum_{i=0}^8 \boldsymbol{\xi}_i f_i \quad (5.5)$$

Applying Chapman–Enskog expansion [97] with respect to Knudsen number ϵ on the particle distribution function leads to:

$$f_i = f_i^{(0)} + \epsilon f_i^{(1)} + \epsilon^2 f_i^{(2)} + \dots \quad (5.6)$$

With the zeroth order in ϵ , we can find that $f_i^{eq} = f_i^{(0)}$, which leads to the definition of the non-equilibrium part as the rest of higher order expansion $f_i^{Neq} = f_i - f_i^{eq}$ [60, 61]. The regularized BGK model aims to recover the macroscopic shear stress by maintaining the second

order in the Knudsen number and assuming that the second-order non-equilibrium function can represent their highest order ($\epsilon f_i^{(1)} \approx f_i - f_i^{(0)}$) [60, 61]. This second order of the non-equilibrium part can be reconstructed using the non-equilibrium momentum flux tensor $\mathbf{\Pi}^{Neq}$, which equals $\mathbf{\Pi}^{Neq} = \mathbf{\Pi} - \mathbf{\Pi}^{eq}$. The momentum flux is obtained from the second moment of the particle distribution function as

$$\Pi_{\alpha\beta} = \sum_{i=0}^8 \xi_{i\alpha} \xi_{i\beta} f_i, \quad \Pi_{\alpha\beta}^{eq} = \sum_{i=0}^8 \xi_{i\alpha} \xi_{i\beta} f_i^{eq}, \quad (5.7)$$

Where f_i^{eq} is calculated from equation (4.4), and α, β are the possible combination of the two directions x, y each, which covers the notations xx, xy, yx, yy . The non-equilibrium distribution function second-order definition [60] gives

$$f_i^{Neq} \approx f_i^{(1)} = \frac{w_i}{2 c_s^4} \mathcal{H}_i \cdot \mathbf{\Pi}^{Neq} \quad (5.8)$$

After calculating the non-equilibrium part f_i^{Neq} , the regularized particle distribution function can be reconstructed just before the collision step as

$$f_i^{Reg} = f_i^{Neq} + f_i^{eq} \quad (5.9)$$

The Regularized LBM shows a remarkable enhancement in stability and accuracy while only truncating up to the second-order non-equilibrium part, which is computationally inexpensive as it adds few steps of computing the momentum flux tensor and the non-equilibrium distribution function [60].

The recursive regularized LBM benefits appear in the stability and accuracy by adding higher-order terms of the equilibrium and non-equilibrium particle distribution functions instead of the second-order in the regularized LBM [62, 63]. The recursive regularized formulas of the equilibrium and non-equilibrium particle distribution function up to the fourth order can be represented in the fourth order as follows [62]:

$$f_i^{eq} = w_i \rho \left(1 + \frac{\xi_i \cdot \mathbf{u}}{c_s^2} + \frac{1}{2 c_s^4} \mathcal{H}_i^{(2)} : \mathbf{u}\mathbf{u} + \frac{1}{2 c_s^6} \left(\mathcal{H}_{ixxy}^{(3)} u_x^2 u_y + \mathcal{H}_{ixyy}^{(3)} u_y^2 u_x \right) + \frac{1}{4 c_s^8} \mathcal{H}_{ixxyy}^{(4)} u_x^2 u_y^2 \right) \quad (5.10)$$

$$f_i^{Neq} = w_i \left(\frac{1}{2 c_s^4} \mathcal{H}_i^{(2)} : \mathbf{a}_1^{(2)} + \frac{1}{2 c_s^6} \left(\mathcal{H}_{ixxy}^{(3)} a_{1,xy}^{(3)} + \mathcal{H}_{ixyy}^{(3)} a_{1,yy}^{(3)} \right) + \frac{1}{4 c_s^8} \mathcal{H}_{ixxyy}^{(4)} a_{1,xyy}^{(4)} \right) \quad (5.11)$$

Where the second-order non-equilibrium distribution function $\mathbf{a}_1^{(2)}$ is the deviatoric stress tensor $\mathbf{\Pi}^{Neq}$, which could be written as:

$$\mathbf{a}_1^{(2)} = \mathbf{\Pi}^{Neq} = \sum_{i=0}^8 \mathcal{H}_i^{(2)} (f_i - f_i^{eq}) \quad (5.12)$$

The higher-order non-equilibrium distribution functions are obtained from the recursive formula based on the second-order non-equilibrium distribution, leading to a velocity space discretization error of order $\mathcal{O}(Ma^{n+1})$ [62], which equals order five by using fourth-order non-equilibrium distribution as shown in equation (5.11). A description of all parts of the Hermite polynomial and non-equilibrium distribution is written in Appendix Section I.

5.4 The lattice Boltzmann overset method with recursive regularized BGK model

The LB-overset method consists of two grids with different reference frames. The fixed grid has the inertial frame, while the moving boundary is attached to the moving grid. The fixed grid is solved using the standard LBE in equation (4.1). While an additional discrete forcing term $F_i(\mathbf{x}, t)$, so called Guo forcing term [98], is added in the LBE of the moving grid as

$$f_i(\mathbf{x} + \boldsymbol{\xi}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t) + \Omega_i \Delta t + S_i \Delta t \quad (5.13)$$

$$S_i = \left(1 - 0.5 \frac{\Delta t}{\tau} \right) F_i(\mathbf{x}, t) \quad (5.14)$$

This discrete force term $F_i(\mathbf{x}, t)$ contains the effect of the inertial density force $\mathbf{F}(\mathbf{x}, t)$ due to relative motion as

$$F_i(\mathbf{x}, t) = w_i \left(\frac{(\boldsymbol{\xi}_i - \mathbf{u}^c) \cdot \mathbf{F}}{c_s^2} + \frac{(\boldsymbol{\xi}_i \cdot \mathbf{u}^c) (\boldsymbol{\xi}_i \cdot \mathbf{F})}{c_s^4} \right) \quad (5.15)$$

$$\mathbf{F} = \rho \mathbf{a}_{Mov} = -\rho (\dot{\mathbf{V}} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}) + 2\boldsymbol{\omega} \times \mathbf{u}^c + \dot{\boldsymbol{\omega}} \times \mathbf{r}) \quad (5.16)$$

The moving frame acceleration $\mathbf{a}_{Mov}$ includes the linear acceleration $-\dot{\mathbf{V}}$, the centripetal acceleration $-\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})$, Coriolis acceleration $-2\boldsymbol{\omega} \times \mathbf{u}^c$, and Euler acceleration $-\dot{\boldsymbol{\omega}} \times \mathbf{r}$, where $\boldsymbol{\omega}$ is the angular velocity, and $\mathbf{r}$ is the distance vector from the center of rotation to the fluid position measured in the moving grid frame. Moreover, a corrected velocity $\mathbf{u}^c$ is calculated to retrieve second-order accuracy called half-force correction [56] as

$$\rho \mathbf{u}^c(\mathbf{x}, t) = \sum_i \xi_i f_i(\mathbf{x}, t) + \mathbf{F}(\mathbf{x}, t) \Delta t / 2 \quad (5.17)$$

This corrected velocity is used in calculating the inertial density force $\mathbf{F}(\mathbf{x}, t)$ and the equilibrium distribution functions $f_i^{eq}(\rho, \mathbf{u}^c)$ used in Ω_i of equation (5.13). However, the corrected velocity $\mathbf{u}^c(\mathbf{x}, t)$ is a function of the inertial density force $\mathbf{F}(\mathbf{x}, t)$ at the same time step, which leads to an implicit problem. This problem can be solved using the exact solution of the Coriolis acceleration $\frac{d\mathbf{u}}{dt} = -2\boldsymbol{\omega} \times \mathbf{u}$ [99], consequently the density forcing term as:

$$\mathbf{u}^c = \begin{bmatrix} u_x \\ u_y \end{bmatrix}_{t+\frac{\Delta t}{2}} = \begin{bmatrix} \cos(\omega\Delta t/2) & \sin(\omega\Delta t/2) \\ -\sin(\omega\Delta t/2) & \cos(\omega\Delta t/2) \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}_t \quad (5.18)$$

The two grids exchange the macroscopic variables through the fixed-to-moving and moving-to-fixed borders, shown in Figure 5.1. These borders act as boundary conditions for both grids where the particle distribution functions $f_i(\mathbf{x}, t)$ are reconstructed. The reconstruction is based on the interpolation of the macroscopic variables $(\rho, \mathbf{u}, \Pi_i^{Neq})$ at each node in the borders using the surrounding nodes of the other grid. The two communication borders are described in the following sections, showing the procedures and the quantities to be interpolated.

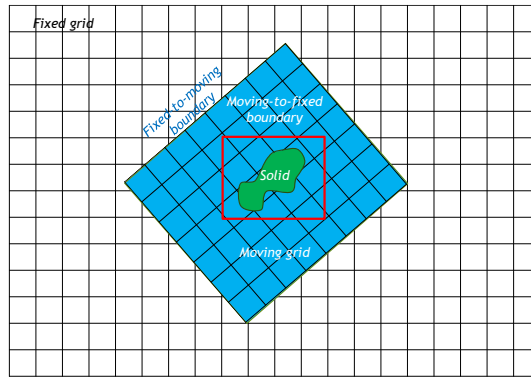


Figure 5.1. The LB-overset method with the illustration of the fixed and moving grids and the communicating borders.

5.4.1 Fixed-to-moving treatment

The fixed-to-moving treatment aims to reconstruct the particle distribution functions on the moving grid borders, which is the fixed-to-moving border. First, on the fixed grid, the macroscopic variables $(\rho, \mathbf{u}, \mathbf{\Pi}^{Neq})$ are interpolated onto the moving grid border projections. This interpolation is implemented using the surrounding nodes as

$$\rho^f(\mathbf{x}^m) = \phi\left(\rho^f(\mathbf{x}^f)\right) \quad (5.19)$$

$$\mathbf{u}^f(\mathbf{x}^m) = \phi\left(\mathbf{u}^f(\mathbf{x}^f)\right) \quad (5.20)$$

$$\mathbf{\Pi}^{Neq,f}(\mathbf{x}^m) = \phi\left(\mathbf{\Pi}^{Neq,f}(\mathbf{x}^f)\right) \quad (5.21)$$

The interpolated function is ϕ , and the superscript (m, f) indicates the variable in the moving and fixed grids, respectively. Then, the velocity vector $\mathbf{u}^f(\mathbf{x}^m)$ and momentum flux tensor $\mathbf{\Pi}^{Neq,f}(\mathbf{x}^m)$ are transformed to the moving grid using the rotation matrix $\mathcal{R}$ as

$$\mathbf{u}_R^f(\mathbf{x}^m) = \mathcal{R} \cdot \mathbf{u}^f(\mathbf{x}^m) \quad (5.22)$$

$$\mathbf{\Pi}_R^{Neq,f}(\mathbf{x}^m) = \mathcal{R} \cdot \mathbf{\Pi}^{Neq,f}(\mathbf{x}^m) \cdot \mathcal{R}^T \quad (5.23)$$

$$\mathcal{R} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \quad (5.24)$$

Then, the velocity is modified by adding the effect of the non-inertial acceleration by subtracting the relative velocity as $\mathbf{u}_{R,A}^f(\mathbf{x}^m) = \mathbf{u}_R^f(\mathbf{x}^m) - \mathbf{v}_r$, where $\mathbf{v}_r = \boldsymbol{\omega} \times \mathbf{r}$, and the subscript (R, A) indicates the modification due to the geometrical rotation and non-inertial acceleration, respectively. In comparison, the non-equilibrium momentum tensor does not need treatments, as the shear stresses are unaffected by the grid rotation [61]. Afterward, the macroscopic variables in the moving grid border are redefined by the modified ones of the fixed grid as

$$\rho^m(\mathbf{x}^m) = \rho^f(\mathbf{x}^m) \quad (5.25)$$

$$\mathbf{u}^m(\mathbf{x}^m) = \mathbf{u}_{R,A}^f(\mathbf{x}^m) \quad (5.26)$$

$$\mathbf{\Pi}^{Neq,m}(\mathbf{x}^m) = \mathbf{\Pi}_{R,A}^{Neq,f}(\mathbf{x}^m) \quad (5.27)$$

Finally, the particle distribution function on the border f_i^m is reconstructed using the redefined macroscopic variables through the reconstruction of the equilibrium and non-equilibrium parts, besides being modified by the half-force correction using the forcing term $F_i(\rho^m(\mathbf{x}^m), \mathbf{u}^m(\mathbf{x}^m))$ to obtain the final value of the reconstructed particle distribution function [56] as

$$f_i^m(\mathbf{x}^m, t) = f_i^{eq}(\rho^m(\mathbf{x}^m), \mathbf{u}^m(\mathbf{x}^m)) - 0.5 F_i(\rho^m(\mathbf{x}^m), \mathbf{u}^m(\mathbf{x}^m)) + f_i^{Neq}(\mathbf{u}^m(\mathbf{x}^m), \Pi^{Neq,m}(\mathbf{x}^m)) \quad (5.28)$$

5.4.2 Moving-to-fixed treatment

The moving-to-fixed communication is implemented with the same previous procedures except adding the relative velocity to the total velocity $\mathbf{u}_A^m(\mathbf{x}^f) = \mathbf{u}^m(\mathbf{x}^f) + \mathbf{v}_r$ instead of subtracting it as in section 5.4.1. This modification is performed before the interpolation and the grid rotation [61]. The reason is that the velocity needs to be treated before being projected in the fixed frame. Moreover, the half-force correction is not required when reconstructing the particle distribution function, as the fixed frame is the inertial reference frame [56].

5.5 Computational Procedures

In this section, the implemented algorithm procedures are discussed:

- 1) Initially, the angular speed ω is defined/updated, and then the rotation angle $\theta = \omega \Delta t$ and angular accelerations $\dot{\omega} = d\omega/dt$ are calculated.
- 2) The new position of the moving grid is updated based on the calculated rotation angle.
- 3) The boundary condition is applied depending on the test case.
- 4) The flow in the fixed and moving grids is solved as follows:
 - a. The density force $\mathbf{F}$, moving grid corrected velocity $\mathbf{u}^c$, and forcing term $F_i(\mathbf{x}, t)$ are calculated using equations in the moving grid (5.15) to (5.18).
 - b. The equilibrium particle distribution function f_i^{eq} is calculated for both grids for all nodes using equation (5.10). The equilibrium particle distribution function for the moving grid is calculated with the corrected velocity $\mathbf{u}^c$.
 - c. The non-equilibrium flux Π^{Neq} is calculated for both grids using equation (5.12).

- d. The non-equilibrium particle distribution function f_i^{Neq} is calculated for both grids using equation (5.11).
 - e. The particle distribution function f_i is reconstructed using the equilibrium and non-equilibrium parts. The fixed grid f_i reconstruction is implemented using equation (5.9), while the moving grid f_i is modified using the half-force correction using equation (5.28).
- 5) The stencil detection algorithm is implemented to detect the surrounding nodes needed for the interpolations described in section 5.6.1.
 - 6) Implement the fixed-to-moving communication starting with interpolating the macroscopic quantities $(\rho, \mathbf{u}, \mathbf{\Pi}^{Neq})$ from the fixed grid to the moving grid nodes located on the fixed-to-moving border using equations (5.19) to (5.21). Then, modify the macroscopic quantities regarding the change of its corresponding frame using equations (5.22) to (5.27). Finally, the particle distribution function is reconstructed on the border the same as 4)a to 4)e.
 - 7) Implement the moving-to-fixed communication starting with interpolating the macroscopic quantities $(\rho, \mathbf{u}, \mathbf{\Pi}^{Neq})$ from the moving grid to the fixed grid nodes located on the moving-to-fixed border using equations (5.19) to (5.21). Then, modify the macroscopic quantities regarding the change of its corresponding frame using equations (5.22) to (5.27). Finally, the particle distribution function is reconstructed on the border the same as 4)b to 4)e.
 - 8) Performing the collision, streaming, and updating the macroscopic variables for both grids, then restarting the procedures from step 1) till the final time step.

5.6 Interpolation algorithm

The interpolator has an important role in the stability and accuracy of the LB-overset method, as its core relies significantly on interpolation. Thus, knowing the best interpolator regarding the LB-overset method is essential. The interpolators compared in the present work are the nearest neighbor, bilinear, radial basis function (RBF), inverse distance weighting (IDW), convolution bicubic, and full-bicubic interpolators. The interpolation algorithm is divided into two main parts: the stencil detection algorithm and the interpolators. The stencil detection algorithm is described

in section 5.6.1, which detects the index of the interpolation-required nodes. Moreover, the interpolators' equations and implementations are illustrated in section 5.6.2.

5.6.1 Stencil detection algorithm

Depending on the needed points count, the stencil detection algorithm detects the correct stencil for each interpolator. For instance, the bicubic interpolator requires 16 surrounding points, while the bilinear interpolator requires 4 points. First, for each node on the moving-to-fixed border, a winding number algorithm [88] is performed using four-point stencils of the moving grid. Next, these stencils are indexed with the first node P_0 in each stencil, allowing us to detect the correct stencil despite the moving grid rotation angle. Then, the algorithm detects the required surrounding nodes based on that indexing, as shown in Figure 5.2. Finally, the needed surrounding variables are extracted using these indices and imported to the interpolators.

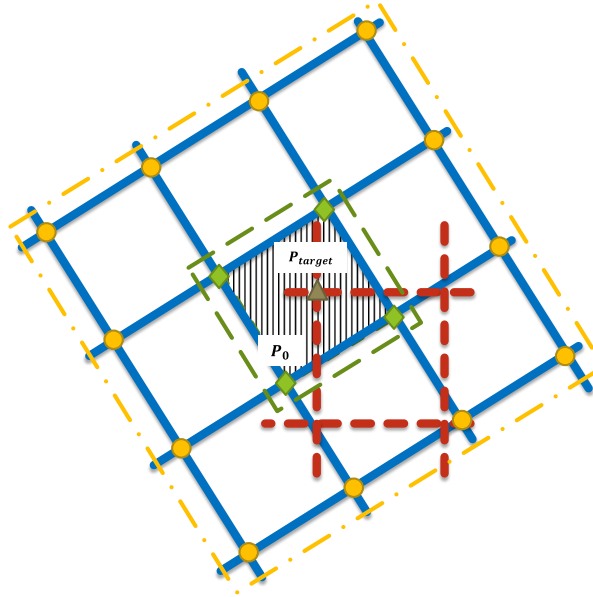


Figure 5.2. Schematic for moving-to-fixed borders stencil detection algorithm, showing the four nodes stencil (the green long-dashed line containing green diamonds), the sixteen nodes stencil (orange dashed line containing the green diamonds and orange circles), the target node located on the border (triangle), and winding number algorithm stencil (black dashed stencil).

5.6.2 Interpolators

This section presents the nearest neighbor, IDW [100], RBF [101], bilinear, bicubic-convolution [102], and bicubic interpolators. These interpolators are selected to cover many categories, such as the number of nodes (like four or sixteen), order of accuracy, and mesh or meshfree interpolators. Moreover, the indexing used for four-points and sixteen points interpolators

is shown in Figure 5.3. Furthermore, all introduced interpolators are verified by retrieving a machine's accuracy for the L2 error norm when using a test function that represents the interpolators' polynomial or equation; Appendix-(II) provides a detailed description of the verification process.

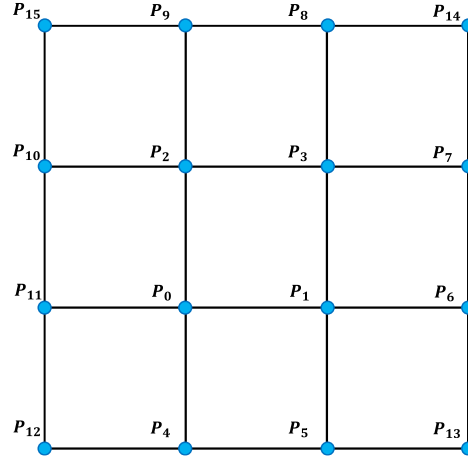


Figure 5.3. Schematic showing the indexing defined for all interpolators.

5.6.2.1 Nearest neighbor

The nearest neighbor could be considered a piecewise constant interpolator that obtains the values from the closest node to the target node. Its algorithm is straightforward to implement and almost does not consume computational resources. The first step in the algorithm is to calculate the distance between the target node and its four surrounding points. Next, detect the minimum distance with its index. Finally, reconstruct the target node macroscopic values with the minimum-distance node values. The nearest neighbor algorithm is selected only to give a reference benchmark.

5.6.2.2 Bilinear interpolator

The nearest neighbor The bilinear interpolator in the present work is based on the polynomial fit of four nodes stencil, which could be written as

$$\phi(x, y) \approx \sum_{m=0}^1 \sum_{n=0}^1 a_{mn} x^m y^n \quad (5.29)$$

Where $\phi(x, y)$ is the interpolated value, and a_{mn} is the polynomial coefficient, which can be calculated by solving the system of linear equations as

$$\begin{bmatrix} 1 & x_0 & y_0 & x_0 y_0 \\ 1 & x_1 & y_1 & x_1 y_1 \\ 1 & x_2 & y_2 & x_2 y_2 \\ 1 & x_3 & y_3 & x_3 y_3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \varphi(x_0, y_0) \\ \varphi(x_1, y_1) \\ \varphi(x_2, y_2) \\ \varphi(x_3, y_3) \end{bmatrix} \quad (5.30)$$

Where $\varphi(x_0, y_0)$, $\varphi(x_1, y_1)$, $\varphi(x_2, y_2)$, $\varphi(x_3, y_3)$ are the field values on the four points P_0, P_1, P_2, P_3 , respectively. However, solving the system of linear equations is performance costly for each interpolation at each time step and should be avoided. Thus, a normalization process is required before the interpolation. This normalization process is implemented by setting the P_0 to a zero position (0,0), then rotating the stencil to a zero angle, which is evaluated by subtracting the position of all the stencil nodes from (x_0, y_0) , then rotating it to a zero-rotational angle using the rotational matrix $\mathbf{R}$. This normalization scheme significantly affects the code performance by decreasing the processing time required for the interpolation, as the stencil position and the inverse matrix are calculated once during the preprocessing and do not need to be calculated each time step. This normalization leads to a more straightforward formula as

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \frac{1}{(\hat{x}_3 - \hat{x}_0)(\hat{y}_3 - \hat{y}_0)} \begin{bmatrix} \hat{x}_3 \hat{y}_3 & -\hat{x}_0 \hat{y}_3 & -\hat{x}_3 \hat{y}_0 & \hat{x}_0 \hat{y}_0 \\ -\hat{x}_3 & \hat{x}_3 & \hat{x}_0 & -\hat{x}_0 \\ -\hat{y}_3 & \hat{y}_0 & \hat{y}_3 & \hat{y}_0 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \varphi(x_0, y_0) \\ \varphi(x_1, y_1) \\ \varphi(x_2, y_2) \\ \varphi(x_3, y_3) \end{bmatrix} \quad (5.31)$$

This formula can be applied to rectangular or square cells, which would be compatible with non-uniform orthogonal grids. Finally, the bilinear interpolator is selected because it has many advantages, such as a straightforward algorithm, low computational resources as it needs only four nodes, second-order accuracy in space [61], and adaptability with non-uniform orthogonal grids.

5.6.2.3 Inverse distance weighting interpolator

The inverse distance weighting is a stochastic multivariate interpolator based on calculating the target node values using the weighted average of the neighboring sample set [100]. The interpolator is presented as

$$\phi(\mathbf{x}) = \begin{cases} \frac{\sum_i^N a_i \varphi_i}{\sum_i^N a_i} & \text{if } d(\mathbf{x}, \mathbf{x}_i) \neq 0 \\ \varphi_i & \text{if } d(\mathbf{x}, \mathbf{x}_i) = 0 \end{cases} \quad (5.32)$$

Where $d(\mathbf{x}, \mathbf{x}_i)$ is the Euclidian displacement between the target point and the nodes and a_i is the weighting coefficient, which equals

$$a_i = d(\mathbf{x}, \mathbf{x}_i)^{-P} \quad (5.33)$$

The power parameter P determines the influence of the set nodes where the increase of P , increases the effect of the nearest nodes. The power parameter cannot exceed the domain dimensions, i.e., $P \leq 2$ in a 2D domain. The power parameter is selected as 2 in the present study, increasing the influence of the nearest nodes to the target point. The IDW is chosen for two main reasons: its simple algorithm and adaptability to any number of nodes, allowing us to compare different node numbers with the same algorithm.

5.6.2.4 Radial basis function interpolator

The radial basis function interpolator is one of the multivariate interpolators widely used in mesh-free methods, which is based on a specific basis function and shape parameter ε . Many basis functions exist, such as Gaussian, multi-quadratic, inverse multi-quadratic, and Poisson RBFs [101]. The shape parameter affects the participation of the radial basis; in other words, the smaller the shape parameter, the more flatted the basis function, and the larger the shape parameter, the more peaked the basis function [101]. The interpolated value is calculated as

$$\phi(x) = \sum_i^n a_i \varphi_i(\|\mathbf{x} - \mathbf{x}_i\|) \quad (5.34)$$

Where the coefficients a_i can be calculated from

$$\boldsymbol{\psi}(\|\mathbf{x}_j - \mathbf{x}_i\|) \cdot \mathbf{a} = \begin{bmatrix} \psi(\|\mathbf{x}_0 - \mathbf{x}_0\|) & \cdots & \psi(\|\mathbf{x}_n - \mathbf{x}_0\|) \\ \vdots & \ddots & \vdots \\ \psi(\|\mathbf{x}_0 - \mathbf{x}_n\|) & \cdots & \psi(\|\mathbf{x}_n - \mathbf{x}_n\|) \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \varphi_0 \\ \vdots \\ \varphi_n \end{bmatrix} \quad (5.35)$$

The calculation of the coefficients a_i requires the inverse radial basis function $\boldsymbol{\psi}^{-1}(\|\mathbf{x}_j - \mathbf{x}_i\|)$ at each time step. However, performing the normalization technique described in section 5.2.2 let it be calculated once in the preprocessing. The selected radial basis function $\boldsymbol{\psi}(\|\mathbf{x}_j - \mathbf{x}_i\|)$ is the inverse multi-quadratic as:

$$\psi_{ij}(\|\mathbf{x}_j - \mathbf{x}_i\|) = (1 + \varepsilon^2 \|\mathbf{x}_j - \mathbf{x}_i\|^2)^{-\frac{1}{2}} \quad (5.36)$$

The RBF is selected because of its adaptability to any number of nodes, allowing us to compare different node numbers with the same algorithm.

5.6.2.5 Bicubic convolution interpolator

The bicubic convolution algorithm solves the bicubic spline interpolation for equally spaced discrete data through an interpolation kernel $\kappa(x)$. This kernel converts the discrete data into continuous functions like the convolution process implementation [102]. The kernel used in this study is introduced by Robert G. Keys [102], which leads to a third-order accuracy as

$$\kappa(x) = \begin{cases} \frac{3}{2} |x|^3 - \frac{5}{2} |x|^2 + 1 & 0 < |x| < 1 \\ \frac{-1}{2} |x|^3 + \frac{5}{2} |x|^2 - 4 |x| + 2 & 1 < |x| < 2 \\ 0 & 2 < |x| \end{cases} \quad (5.37)$$

This kernel leads to a more friendly matrix notation in 1D as

$$\phi(x) = 0.5 \begin{bmatrix} 1 & dx & dx^2 & dx^3 \end{bmatrix} \begin{bmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \phi_{-1} \\ \phi_0 \\ \phi_1 \\ \phi_2 \end{bmatrix} \quad (5.38)$$

In 2D, equation (5.38) is solved four times in the x-direction at the projection of the target point to have four interpolated values for each row, the diamond symbols in Figure 5.4. Then, these four interpolated values are interpolated again in the y-direction at the location of the target node, as shown in Figure 5.4. For instance, the first row contains the points P_{12}, P_4, P_5, P_{13} , with corresponding function values $\phi_{-1}, \phi_0, \phi_1, \phi_2$, respectively. Then, this interpolated value of the first node is used as ϕ_{-1} for the interpolation in the y-direction.

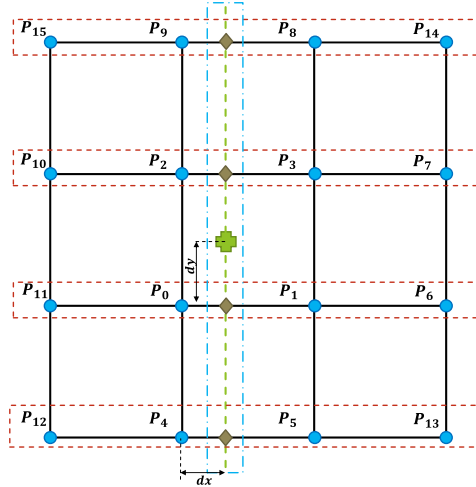


Figure 5.4. Schematic showing the four interpolations in the x-direction (four red dashed stencils) implemented at the target node projection (four diamonds) and the interpolation in the y-direction (cyan dot-dashed stencil) implementing the total interpolation on the target node (green cross).

It is essential to mention that the bicubic convolution interpolator is an equally spaced non-rotated interpolator; thus, a normalization process must be implemented before the interpolation,

as described previously in the bilinear interpolator section. Moreover, It has a good combination of computational cost and accuracy [103], and its equally spaced requirement leads to being widely used in real-time imaging processing. Thus, the bicubic convolution interpolation could be suitable with LBM as the standard LBM has an equally spaced grid.

5.6.2.6 Bicubic interpolator

The bicubic interpolator is a third-order interpolator that calculates the target node value using 16 surrounding points using the following equation:

$$\phi(x, y) = \sum_{i=0}^3 \sum_{j=0}^3 a_n x^i y^j \quad (5.39)$$

The equation contains 16 unknown coefficients a_n , which can be calculated by solving the following system of linear equations as

$$\mathbf{A} \cdot \mathbf{a} = \begin{bmatrix} 1 & x_0 & y_0 & \cdots & x_0^3 y_0^2 & x_0^2 y_0^3 & x_0^3 y_0^3 \\ & \vdots & & \ddots & & \vdots & \\ 1 & x_{15} & y_{15} & \cdots & x_{15}^3 y_{15}^2 & x_{15}^2 y_{15}^3 & x_{15}^3 y_{15}^3 \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_{15} \end{bmatrix} = \begin{bmatrix} \phi_0 \\ \vdots \\ \phi_{15} \end{bmatrix} \quad (5.40)$$

This formula is performance costly as it is required to solve 16 linear equations, which could be enhanced by implementing the normalization process. The normalization allows the calculation of the LHS matrix $\mathbf{A}$ in the preprocessing, leading to the evaluation of the 16 coefficients using the inverse matrix formula as $\mathbf{a} = \mathbf{A}^{-1} \boldsymbol{\phi}$. The bicubic interpolator is chosen because it has a third-order accuracy in space; thus, it is expected to have the most accurate results. Moreover, it is compared with straightforward algorithms, such as IDW and the bicubic convolution algorithm, to indicate the interpolator efficiency. In other words, see whether a simple algorithm could give the same results as the complex ones in our test cases.

5.7 The result and discussion

This section has five test cases to show the interpolator's effect with increasing flow complexity. First, section 5.7.1 investigates the uniform flow to verify the integration of the interpolators with the LB-overset method and detect any slight distortion to the flow. Then, in section 5.7.2, the Poiseuille flow is implemented to add more complexity to the velocity field. Next, section 5.7.3 solves the Taylor-Couette flow to test the boundaries rotation effect with a complex steep velocity profile. After that, in section 5.7.4, a vortex advection is solved to

investigate the interpolators in unsteady flow with a high steep density profile. Finally, section 5.7.5 discusses the uniform flow around a rotating cylinder to validate the interpolators with the overset method. In this study, the system of units is the LB system of units, so the space and time steps are scaled to unity, i.e., $\Delta x = \Delta y = \Delta t = 1$.

5.7.1 Uniform flow

The uniform flow is implemented to detect any slight disturbance or deformation on the flow field due to the motion of the moving grid. Moreover, it acts as a verification step for integrating the interpolators' algorithms with the LB-overset method. The flow field boundary conditions are velocity inlet and pressure outlet evaluated using the Zou and He boundary condition [85], while the lower and upper surfaces are straight velocities implemented according to the Inamuro boundary condition [87]. The simulation parameters for all interpolators are the fixed grid size of 128×64 cells, relaxation time $\tau = 0.7$, Reynolds number $Re = \frac{u_\infty L}{\nu} = 100$, reference length $L = N_y^f = 64$, kinematic viscosity $\nu = \frac{1}{3}(\tau - 0.5)$, Mach number $Ma = u_{max}/c_s = 0.1776$, rotational Mach number $Ma_{rot} = r \omega / c_s = 0.001385$, the moving grid's half-diagonal r , and the fixed-to-moving grid size ratio of $N_y^f / N_y^m = 4$.

The simulation starts with zero velocity and unity density in the overall domain. Then, the flow is solved without rotating the moving grid till the steady state time-step $t_s = 10000$ time-step, which ensures that the flow reaches the steady state and the normalized L_1 velocity discrepancy L_1^u reaches the machine's accuracy. The discrepancy L_1^u is calculated as

$$L_1^u = \frac{1}{N} \sum_{k=0}^{N-1} \frac{\|\mathbf{u}(\mathbf{x}_k) - \mathbf{u}_{ref}(\mathbf{x}_k)\|}{\max(\mathbf{u}_{ref})} \quad (5.41)$$

Where the reference velocity $\mathbf{u}_{ref}$ is the velocity of the same test case with the fixed grid only, the nodes' index is k , and N is the total number of nodes. After that, the moving grid rotates for $t_r = 50000$ more time-step, which covers more than one complete cycle of the moving grid around its center. Moreover, Figure 5.5 shows the velocity field of the two overlapped grids at the time $t_f = t_s + t_r$, for the bicubic interpolator as an example, with almost no flow distortion.

The interpolators are tested regarding the discrepancy L_1^u , its time-average value $\hat{L}_1^u$, the pressure noise at the final time $\frac{\partial P}{\partial t}\big|_{t_f}$, and its space-average value $\frac{\partial \bar{P}}{\partial t}$ as

$$\hat{L}_1^u = \frac{1}{(t_f - t_s)} \sum_{t=t_s}^{t_f} L_1^u(t) \quad (5.42)$$

$$\frac{\partial P}{\partial t}\big|_{t_f} = \frac{P_{t_f} - P_{t_f - \Delta t}}{\Delta t} \quad (5.43)$$

$$\frac{\partial \bar{P}}{\partial t} = \frac{1}{N} \sum_{k=0}^{N-1} \frac{\partial P}{\partial t}\big|_{t_f, k} \quad (5.44)$$

The discrepancy L_1^u is calculated to evaluate the effect of the interpolators on the velocity field while the grid rotates. The average-time value indicates the interpolators' performance over time after the grid motion. The pressure noise indicates the noise accompanied by the particle distribution function reconstruction on the border during the rotation. Moreover, the particle distribution function directly affects the pressure calculation on each node. Thus, the pressure change with time indicates the accuracy of particle distribution function reconstruction and the interpolators' effects in consequence.

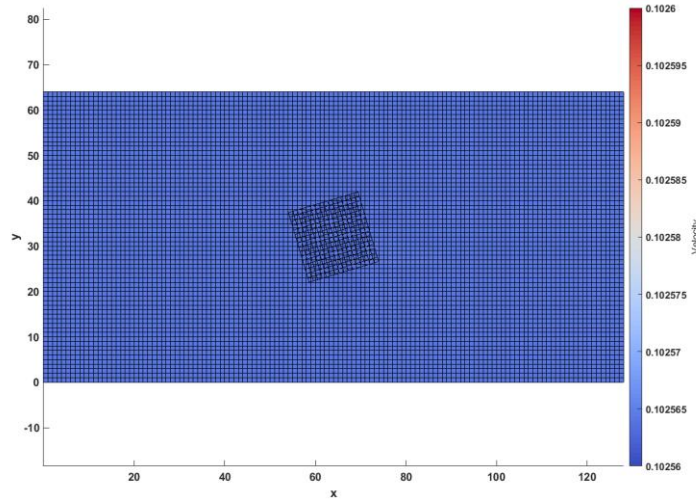


Figure 5.5. The uniform flow velocity field is shown on the overlapped grids implemented at t_f for the bicubic interpolator.

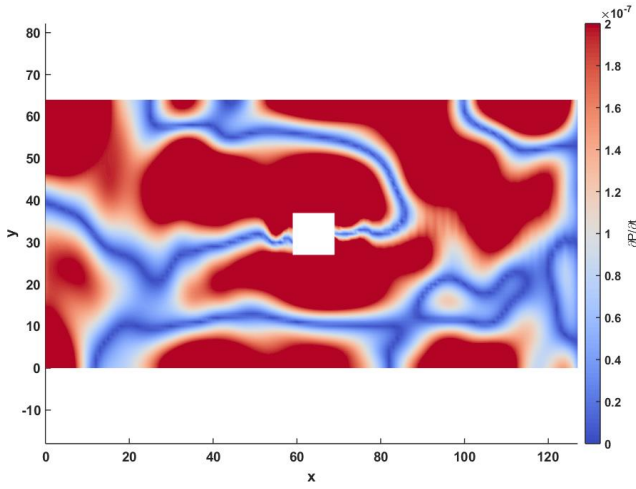
Before the moving grid started to rotate, all interpolators had the machine accuracy ($L_1^u < 10^{-14}$), which verifies that the interpolators had integrated correctly with the LB-overset. Table 5.1 shows L_1^u , $\hat{L}_1^u$, and $\frac{\partial \bar{P}}{\partial t}$ after the grid rotation. The nearest neighbor interpolator is evaluated as

the benchmark of this test case, which is expected to have a high error and noise value for any general case. Moreover, the RBF16 and RBF4 interpolators with the shape parameter $\varepsilon = 1.0$ gave high error and noise values after the rotation, higher than the nearest neighbor. An explanation might be the sum of the coefficients $a_0 + a_1 + \dots + a_n \neq 1$. When the shape parameter is selected as $\varepsilon = 0.1$, the noise due to the rotation is almost suppressed, with a slightly high value for the velocity error. Furthermore, the IDW4 and IDW16 interpolators have acceptable results with the same order of magnitude, indicating almost no difference regarding the IDW interpolator nodes number. Finally, the bicubic, bicubic convolution algorithm, and bilinear interpolators have the same order of magnitude for each metric and are the lowest values between the interpolators.

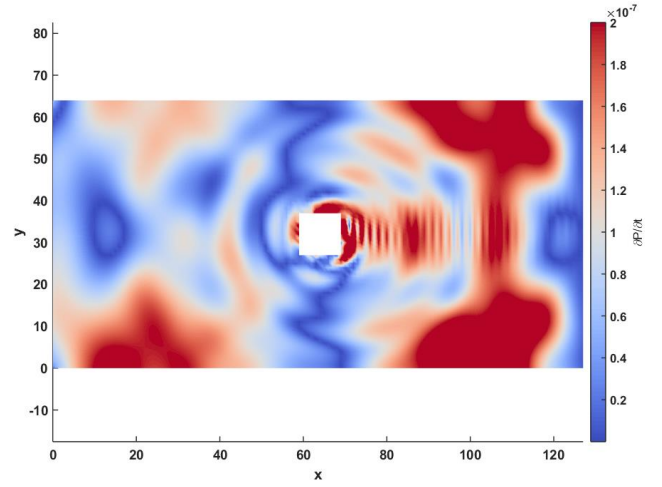
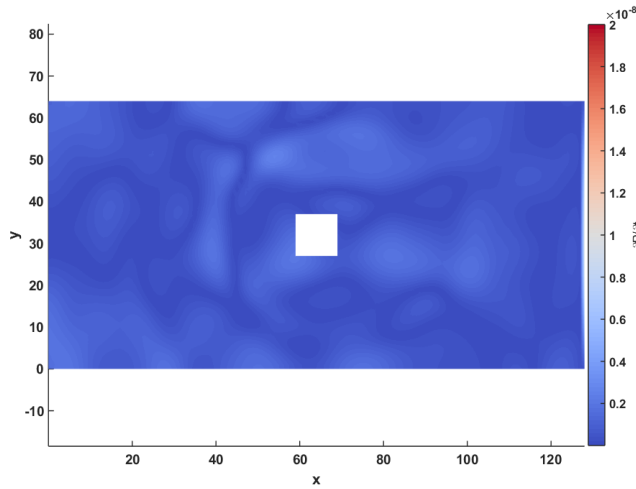
Table 5.1. The L_1 norm error and pressure noise for the introduced interpolators after grid rotation at t_f .

Interpolator	L_1^u	$\hat{L}_1^u$	$\frac{\overline{\partial P}}{\partial t}$
Nearest neighbor	1.1519e-07	2.2345e-08	1.8423e-09
RBF4, $\varepsilon = 1.0$	9.0222e-03	4.9615e-03	3.2689e-05
RBF16, $\varepsilon = 1.0$	9.6981e-04	6.2939e-04	3.6991e-07
RBF16, $\varepsilon = 0.1$	1.8330e-06	2.8948e-06	1.1385e-09
IDW4	2.8872e-08	1.1385e-09	6.8323e-10
IDW16	5.8952e-09	4.5429e-09	6.5709e-12
Bilinear	1.0930e-08	7.9846e-09	4.2434e-12
Bicubic convolution	5.8622e-09	4.5465e-09	4.1673e-12
Bicubic	5.1911e-09	4.4038e-09	4.9417e-12

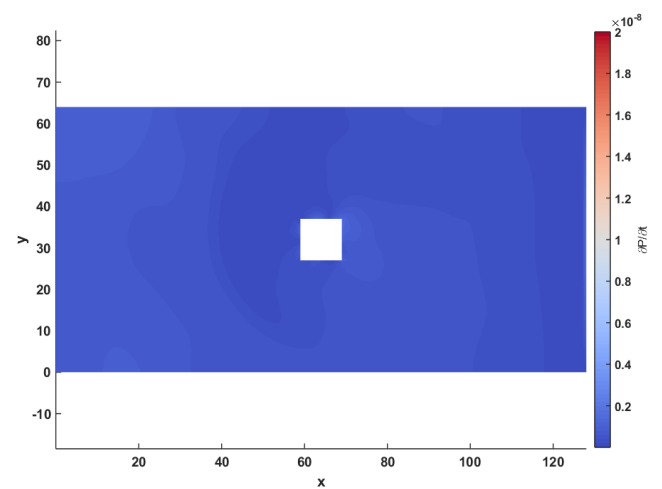
Visualizing the pressure noise field over the fixed grid illustrates the interpolators' effect on the simulations. First, The benchmark interpolator, i.e., the nearest neighbor, has relatively high-pressure noise expanding over the flow field, as shown in Figure 5.6(a). Then, the RBF16 with shape parameter $\varepsilon = 0.1$ shows a high pressure noise originated in the vicinity of the border and spreading over the entire flow field on a pattern similar to a capillary wave, as shown in Figure 5.6(b). After that, the IDW4 and IDW16 have almost the same pressure noise pattern. Moreover, the pressure noise is almost suppressed near the border, with a low-pressure noise spreading over the flow field, as shown in Figure 5.6(c). Finally, the bicubic, bicubic convolution, and bilinear interpolators have almost the same pressure noise field. Furthermore, these interpolators significantly suppress the noise in the vicinity of the border and over the entire flow field, as shown in Figure 5.6(D).



(a) Nearest neighbor.

(b) RBF16 with $\varepsilon = 0.1$. The pressure noise of RBF16 and RBF4 with $\varepsilon = 1$ are much higher and not shown here.

(c) IDW16. The interpolator IDW4 gives a very similar pattern with a higher noise value.



(d) Bilinear. The bicubic and bicubic convolution interpolators give very similar results.

Figure 5.6. The pressure noise $\left. \frac{\partial P}{\partial t} \right|_{t_f}$ showed on the fixed grid after rotation. A factor of 10 is used for the nearest neighbor and RBF16 color bars to visualize the pressure noise field.

These results suggest that the nearest neighbor interpolator is a good choice for integrating with the LB-overset method for uniform flow due to the good pressure noise and velocity error values. Moreover, the RBF4 and RBF16 have a high error regarding the velocity and particle distribution function reconstruction, even with decreasing the shape parameter as the noise waves still exist. Also, decreasing the shape parameter flattened the radial basis function [101], which could lead to the error increment in more complex flow. Thus, the RBF interpolator with up to 16 nodes in the uniform flow is considered not a good choice that needs an optimization function to select the

optimum shape parameter value regarding the specified number of nodes, which requires more investigation and is not covered in this study. Furthermore, the interpolated field quantities are not conserved because the sum of coefficients is not exactly 1. As a result, the RBF4 and RBF16 interpolators will not be investigated more in this work. On the other hand, the nearest neighbor, IDW4, IDW16, bilinear, bicubic convolution, and bicubic interpolators are successfully integrated and adapted to the LB-overset algorithm for uniform flow.

5.7.2 Poiseuille flow

The Poiseuille flow adds more complexity as the effect of the velocity gradient could be investigated on the interpolators. The flow field boundary conditions are velocity inlet and pressure outlet evaluated using the Zou and He, while the far field of the lower and upper surfaces are halfway bounceback [28]. The simulation parameters for all interpolators are the fixed grid size of 128×64 cells, relaxation time $\tau = 0.7$, Reynolds number $Re = \frac{u_\infty L}{\nu} = 100$, reference length $L = H = N_y^f + 1 = 65$, kinematic viscosity $\nu = \frac{1}{3}(\tau - 0.5)$, Mach number $Ma = u_{max}/c_s = 0.1776$, rotational Mach number $Ma_{rot} = r \omega / c_s = 0.001385$, and the fixed to moving grid size ratio of $N_y^f / N_y^m = 4$. The inlet velocity profile is

$$u_{x,inlet} = u_{x,max} \left(\frac{4y}{H} - \frac{4y^2}{H^2} \right) \quad (5.45)$$

The results are tested by calculating the maximum velocity field discrepancy $L_\infty^u = \max \left(\frac{\|u(x_k) - u_{ref}(x_k)\|}{\max(u_{ref})} \right)$, where the reference velocity u_{ref} is the velocity of the same test case with the fixed grid only, the discrepancy L_1^u as in Eqn. (5.41), and the pressure-noise space-average value $\frac{\bar{\partial p}}{\partial t}$ as in Eqn. (5.44). As discussed in section 5.7.1, the discrepancy L_1^u and the pressure noise $\frac{\bar{\partial p}}{\partial t}$ are calculated to indicate the effect of interpolators on the velocity field and particle distribution function reconstruction, respectively. Moreover, the maximum velocity field discrepancy gives an explicit quantity of the velocity field deviations on individual nodes, testing the effect of rotation near its vicinity and not interest in the overall domain.

The simulation starts with zero velocity and unity density in the overall domain. Then, the flow is solved without rotating the moving grid till the steady state time-step $t_s = 20000$ time-step, which ensures that the flow reaches a steady state and the $L_1^u \leq 10^{-7}$ calculated as in Eqn. (5.41),

which corresponds to the time-step t_s . After that, the moving grid rotates for $t_r = 80000$ more time-step, which covers more than one complete cycle of the moving grid around its center. The velocity field of the two overlapped grids at $t_f = t_s + t_r$ in Figure 7 shows the smooth transition of the velocity field between the two grids for the bicubic interpolator as an example; consequently, the macroscopic data are correctly transferred between the grids.

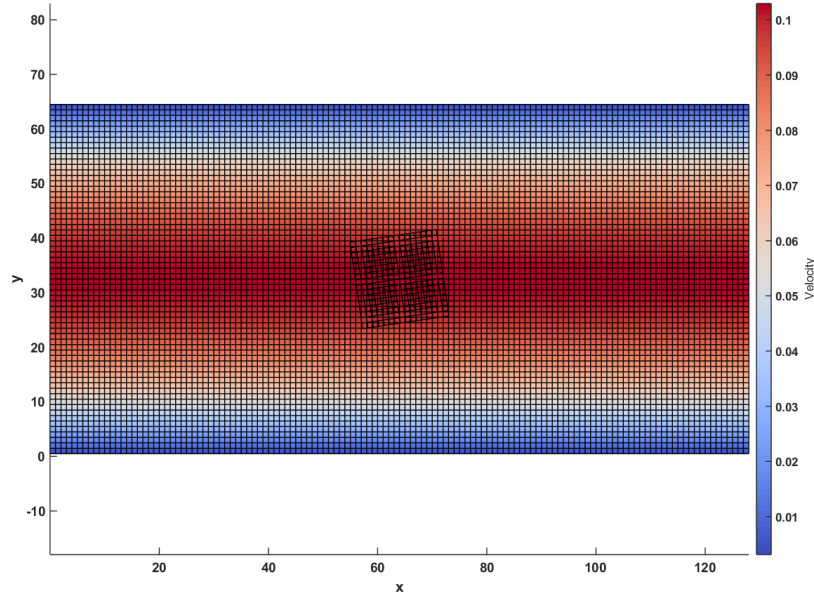
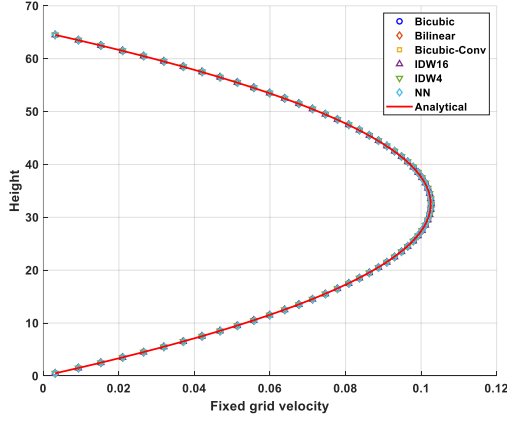
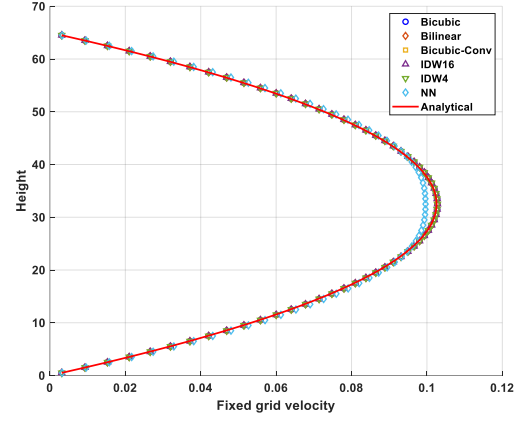


Figure 5.7. An example of the velocity field for the bicubic interpolator at t_f is presented in the overlapped grids.

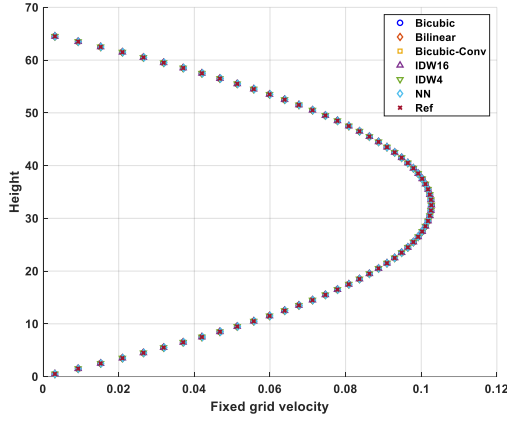
Figure 5.8 shows the fixed grid velocity profile located before and after the moving grid at 30% and 70% of the channel's length for all interpolators. The velocity profiles before and after the moving grid illustrate the effect of passing the data between the moving and fixed grids. As shown in Figure 5.8(b) and (d), the nearest neighbor interpolator shows a high deviation of the velocity profiles from the analytical and numerical solution without overset, especially after the moving grid at 70% of the channel's length. This deviation results from the incorrect velocity field transferred from the moving grid to the fixed one, which affects the velocity profile behind the moving grid. On the other hand, The IDW4, IDW16, bilinear, bicubic convolution, and bicubic interpolators' velocity profiles are identically matched visually with the analytical and numerical solutions without overset. This matching shows that no distortion appears for more complex steady flows, and there is almost no difference between the interpolators on the velocity profile.



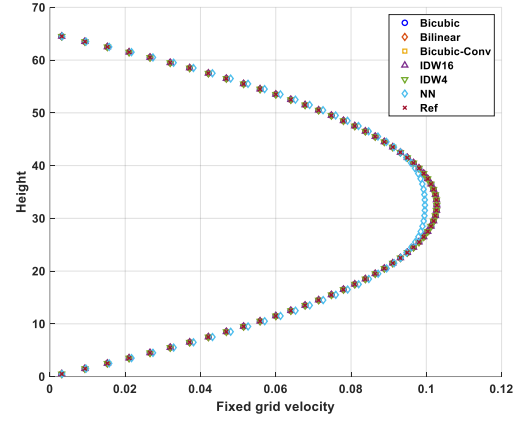
(a) Velocity profile at 30% of the channel length compared with the analytical solution.



(b) Velocity profile at 70% of the channel length compared with the analytical solution.



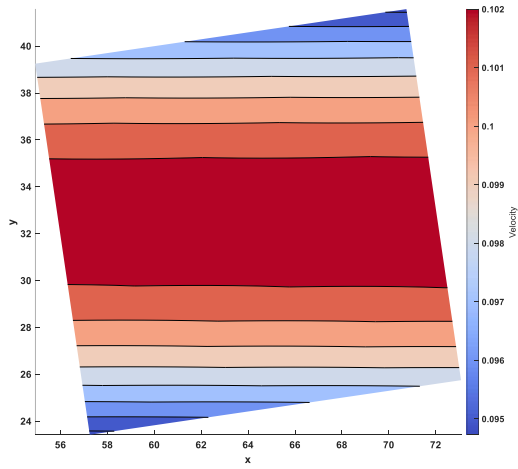
(c) Velocity profile at 30% of the channel length compared with the numerical solution.



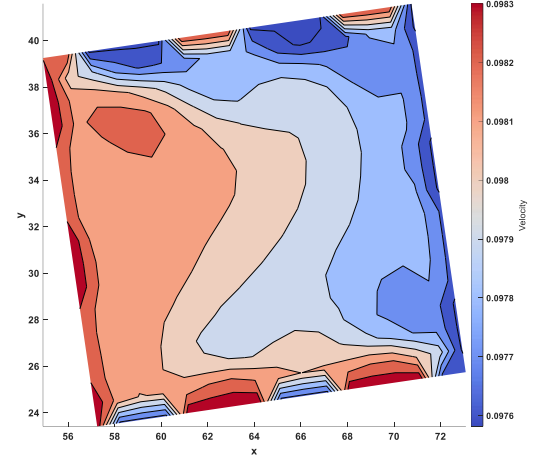
(d) Velocity profile at 70% of the channel length compared with the numerical solution.

Figure 5.8. The velocity profile comparison of the different interpolators before and after the moving grid at t_f .

Moreover, Figure 5.9(a) shows the velocity field contours in the moving grid for the bicubic interpolator as straight lines, indicating that the velocity profile is unaffected by the moving grid rotation. On the other hand, Figure 5.9(b) shows the velocity field contours for the nearest neighbor interpolator, which is quasi-uniform over the moving grid. One of the reasons could be that the nearest neighbor did not capture the velocity gradient in the vertical direction. Consequently, the nearest neighbor interpolator cannot transfer the correct velocity values to the moving grid, resulting in an incorrect velocity profile for the moving grid.



(a) Bicubic interpolator. The interpolators IDW4, IDW16, Bilinear, and bicubic convolution interpolators have the same velocity field.



(b) Nearest neighbor.

Figure 5.9. Moving grid velocity field at t_f for bicubic interpolator (a) and nearest neighbor interpolator (b).

Table 5.2 shows the L_∞^u , L_1^u , and $\frac{\partial \bar{P}}{\partial t}$ for the interpolators. The nearest neighbor has the highest values for each metric, resulting from not capturing the velocity gradient and, consequently, the incorrect velocity profile downstream of the moving grid. Moreover, the other interpolators have the same order of each metric with bicubic convolution, and bicubic interpolators have the lowest values.

Table 5.2. The velocity discrepancy and pressure noise for the introduced interpolators.

Interpolator	L_∞^u	L_1^u	$\frac{\partial \bar{P}}{\partial t}$
Nearest neighbor	4.3122e-02	5.6182e-03	5.3695e-06
IDW4	4.9673e-04	6.8831e-05	1.6365e-07
IDW16	8.8743e-04	7.7527e-05	1.8863e-07
Bilinear	4.4406e-04	6.7249e-05	1.9151e-07
Bicubic convolution	1.2335e-04	4.2033e-05	1.8583e-07
Bicubic	1.2335e-04	4.2033e-05	1.8583e-07

These findings show that the IDW4, IDW16, bilinear, bicubic convolution, and bicubic interpolators perform almost the same for the complex steady flow despite the number of interpolator nodes. Moreover, it was very interesting that the Poiseuille flow does not help to distinguish between the latter interpolators' performance. As a result, the Taylor-Couette flow will

be evaluated to provide a more complex flow, and the nearest neighbor interpolator will not be investigated anymore.

5.7.3 Taylor-Couette flow

The Taylor-Couette flow is implemented to investigate the interpolators' performance with boundary rotation and a steeper velocity profile for a steady solution where the flow is simulated between two cylinders. The inner cylinder is placed inside the moving grid, while the outer cylinder is on the fixed grid, borders 1 and 2 shown in Figure 5.10, respectively. The domain boundaries are the two cylinders with zero velocities imposed using the curved boundary treatment implemented by the Guo extrapolation technique [30]. Consequently, the effect of the inner cylinder rotation appears only from the moving grid rotation. The moving-to-fixed border, border 4, is selected as a circle to avoid the communication borders' rapprochement or intersection. Moreover, the flow in the fixed grid is solved for all nodes between borders 2 and 4. On the other hand, the flow in the moving grid is solved for all nodes located between borders 1 and 3.

The two cylinders' centers are in the center of a 192×192 square, and the outer radius $R_{out} = 0.45H$, where H is the square edge length. Moreover, the inner circle $R_{in} = \beta R_{out}$, where β is the radii ratio, the reference length is the thickness between the two cylinders $L = R_{out} - R_{in}$, relaxation time $\tau = 1.0$, kinematic viscosity $\nu = \frac{1}{3}(\tau - 0.5)$, Reynolds number $Re = \frac{u_{max} L}{\nu} = 10$, the velocity on the inner cylinder $u_{max} = 0.024$, Mach number $Ma = u_{max}/c_s = 0.0417$, the angular speed $\omega = u_{max}/R_{in} = 0.001385$, the fixed-to-moving grid size ratio of $N_y^f/N_y^m = 2$, and the radius of the moving-to-fixed border is twice the inner cylinder radius. The domain is initialized with zero velocity and unity density, with moving grid rotation starting in the first time step till the final time step $t_f = 100000$.

The analytical solution for the Taylor-Couette flow [89] is based on the inner and outer cylinders' radiuses and their corresponding velocities, as shown in equation (5.46). In this test case, the outer cylinder velocity is zero, while the inner has the maximum speed. Thus, the analytical solution could be presented as a function of the radii ratio β as shown in equation (5.47).

$$u_{Ana}(x, y) = u_{inner} \left(\frac{\frac{R_{out}}{r(x, y)} - \frac{r(x, y)}{R_{out}}}{\frac{R_{out}}{R_{in}} - \frac{R_{in}}{R_{out}}} \right) + u_{outer} \left(\frac{\frac{r(x, y)}{R_{in}} - \frac{R_{in}}{r(x, y)}}{\frac{R_{out}}{R_{in}} - \frac{R_{in}}{R_{out}}} \right) \quad (5.46)$$

$$u_{Ana}(x, y) = u_{max} \left(\frac{\beta}{1 - \beta^2} \right) \left(\frac{R_{out}}{r(x, y)} - \frac{r(x, y)}{R_{out}} \right) \quad (5.47)$$

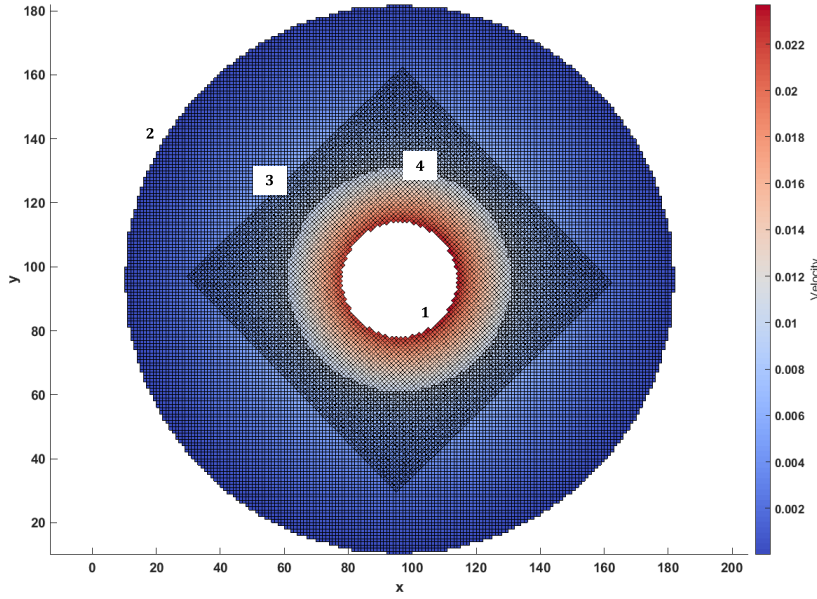


Figure 5.10. An example of the velocity field for the bilinear interpolator is presented in the overlapped grids. Border No. 1 is the inner cylinder boundary condition located on the moving grid, border No. 2 is the outer cylinder situated in the fixed grid, border No.3 is the fixed-to-moving border, and border No. 4 is the moving-to-fixed border.

The analytical solution shows that the lower the radii ratio β , the steeper the velocity profile; thus, the radii ratio $\beta = 0.2$ is selected. Moreover, the L1 velocity error e_1^u , and maximum absolute velocity error e_{max}^u are calculated to be metrics for the interpolators' comparison as

$$e_1^u = \frac{1}{N} \sum_{k=0}^{N-1} |\mathbf{u}(\mathbf{x}_k) - \mathbf{u}_{ana}(\mathbf{x}_k)| \quad (5.48)$$

$$e_{max}^u = \max(|\mathbf{u}(\mathbf{x}_k) - \mathbf{u}_{ana}(\mathbf{x}_k)|) \quad (5.49)$$

Where $\mathbf{u}_{ana}$ is the velocity field analytical solution. The errors are calculated in both grids as the moving grid significantly affects the velocity profile. Moreover, k is the node index

corresponding to each grid with its corresponding length N . Figure 5.11 shows the velocity profile for the bilinear interpolator, as an example, in the fixed and moving grids with the analytical solution. The numerical and analytical profiles are almost identical. Moreover, the overlapped areas of the fixed and moving grids coincide, as shown in Figure 5.11(b), revealing the perfect data communication between the grids on the borders.

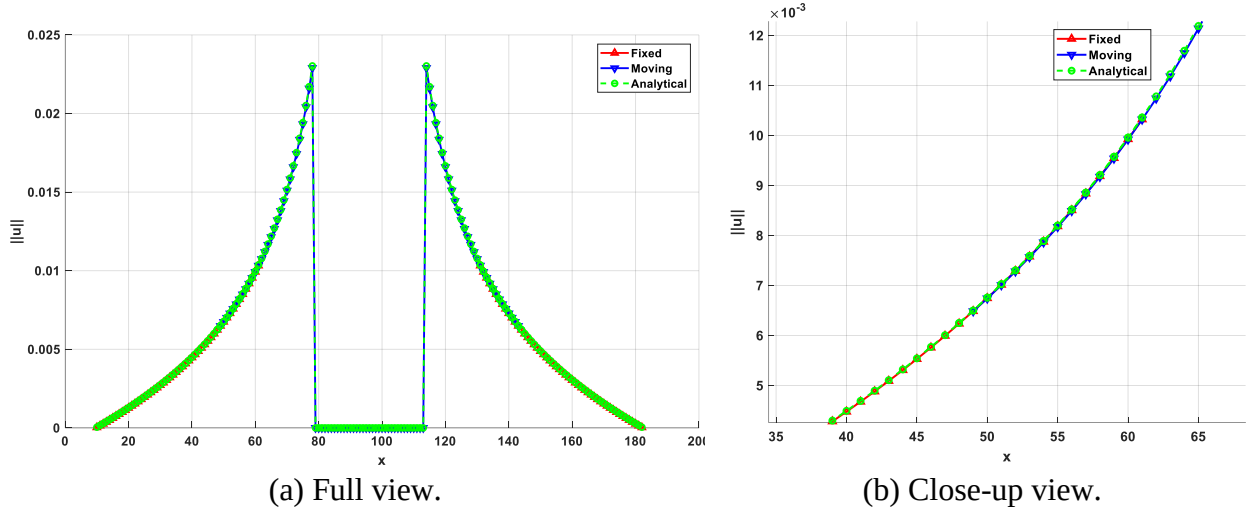


Figure 5.11. The velocity profile for the bilinear interpolator at t_f .

Table 5.3 shows the e_1^u and e_{max}^u for the interpolators. The introduced interpolators have almost the same values for each metric, with the bicubic interpolator having the lowest values.

Table 5.3. The velocity error for the introduced interpolators.

Interpolator	e_1^u	e_{max}^u
IDW4	5.7928e-05	1.5396e-04
IDW16	5.7770e-05	1.5387e-04
Bilinear	5.8285e-05	1.5385e-04
Bicubic convolution	5.7939e-05	1.5379e-04
Bicubic	5.7332e-05	1.5371e-04

These findings show that the IDW4, IDW16, bilinear, bicubic convolution, and bicubic interpolators behave similarly for the Taylor-Couette flow. Again, the steady flows could not distinguish between the interpolators. Thus, the unsteady flow is investigated in the next section using the advection vortex in a uniform flow.

5.7.4 High Reynolds advection vortex in uniform flow

The advection vortex tests the interpolators' behavior with time through its unsteady solution with a highly steep density profile. The used vortex model is called the barotropic vortex model, which has a good consistency with athermal LBM [104]. The velocity and density of the vortex [61, 104] are defined as

$$u_x = u_\infty - \varepsilon_{vor} \left(\frac{y - y_c}{R_{vor}} \right) \exp \left(- \frac{(x - x_c)^2 + (y - y_c)^2}{2 R_{vor}^2} \right) \quad (5.50)$$

$$u_y = \varepsilon_{vor} \left(\frac{x - x_c}{R_{vor}} \right) \exp \left(- \frac{(x - x_c)^2 + (y - y_c)^2}{2 R_{vor}^2} \right) \quad (5.51)$$

$$\rho = \rho_0 \exp \left[- \frac{\varepsilon_{vor}^2}{2c_s^2} \exp \left(- \frac{(x - x_c)^2 + (y - y_c)^2}{R_{vor}^2} \right) \right] \quad (5.52)$$

Where, ε_{vor} is the vortex strength, the vortex radius R_{vor} , and (x_c, y_c) is the vortex center location. The fixed grid is selected as a 200×200 square with a fixed-to-moving grid size ratio of $N_y^f/N_y^m = 3$. A high Reynolds number $Re = \frac{2 u_\infty R_{vor}}{\nu} = 10^4$ is used to prevent the dissipation of the vortex through time as it leads to significantly low kinematic viscosity. Moreover, decreasing the vortex radius reduces the kinematic viscosity, enhancing the vortex consistency besides resulting in a higher steep density profile; thus, the vortex radius is selected as $R_{vor} = N_y^f/20$. The free stream velocity is calculated from the Mach number $u_0 = Ma c_s$, Mach number $Ma = 0.17$, and the relaxation time is calculated from $\tau = 3\nu + 0.5 \approx 0.5006$.

The domain is initialized in both grids using equations (51) to (53), with the moving grid rotating from the first time step. Moreover, the particle distribution function is initialized using its equilibrium part calculated from equation (5.10). Furthermore, the vortex is located in front of the moving grid at $x_c = N_x/5$, as shown in Figure 5.12. Then, the vortex starts to move with time due to the presence of the free stream velocity in equation (4.30) passing through the moving grid. A periodic boundary condition is applied on the fixed grid square boundaries to let the vortex re-enter the domain. The simulation is continued for five flow-through-time (FTT), which means that the vortex passed through its initial location 5 times, which corresponds to $t_f = 5 N_x^f/u_\infty = 10188$. Finally, a reference test case is solved without overset method in one fixed grid, with the same initialization and simulation parameters.

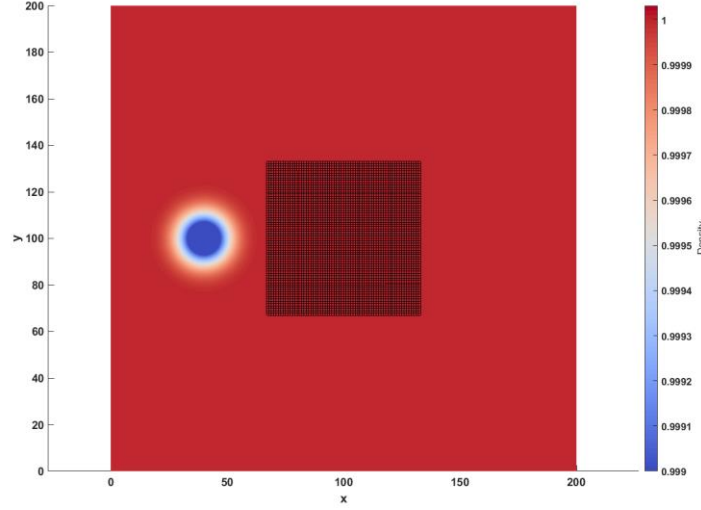


Figure 5.12. The initial condition of the density field is shown in the two grids.

Figure 5.13 shows the density profile of the vortex after five FTTs. The bicubic and bicubic convolution interpolators have conserved the same vortex density profile compared to the reference test case. On the other hand, the IDW4, IDW16, and bilinear interpolators have more deviations.

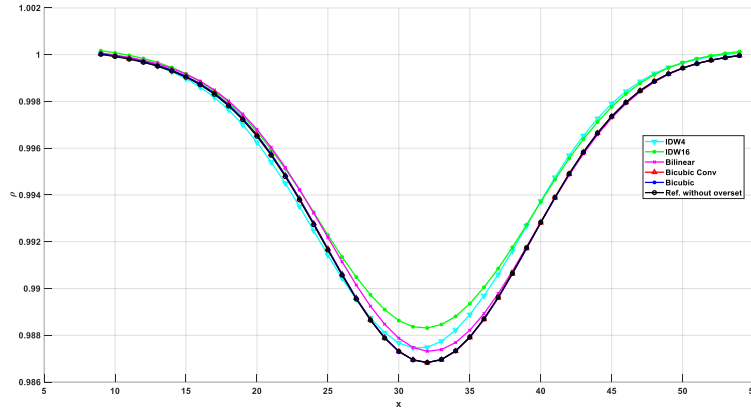


Figure 5.13. The vortex density profile for different interpolators and the reference test case (without overset) at FTT=5.

Moreover, the L1 density discrepancy L_1^ρ and the maximum absolute discrepancy L_{max}^ρ are calculated as comparison metrics for the interpolators as

$$L_1^\rho = \frac{\sum_{k=0}^N \rho_k - \rho_k^{ref}}{\sum_{k=0}^N \rho_k^{ref}} \quad (5.53)$$

$$L_{max}^\rho = \max(|\rho_k - \rho_k^{ref}|) \quad (5.54)$$

Table 5.4 shows that the IDW4 and IDW16 have a markable high discrepancy than the other interpolators for each metric. Moreover, the bilinear has higher discrepancies than the bicubic and bicubic convolution interpolators. Furthermore, the bicubic convolution and the bicubic interpolators have similar discrepancies, with the bicubic interpolator having the lowest values for each metric.

Table 5.4. The density discrepancy for the introduced interpolators at FTT=5.

Interpolator	L_1^p	L_{max}^p
IDW4	1.1226e-04	1.4184e-03
IDW16	1.1561e-04	1.7889e-03
Bilinear	7.8487e-06	6.6769e-04
Bicubic convolution	3.4217e-06	7.5465e-05
Bicubic	3.1500e-06	7.5411e-05

These findings show that the IDW4, IDW16, and bilinear interpolators would be more exposed to the unsteady pressure noise and could not conserve the vortex density profile. On the other hand, the bicubic and bicubic convolution preserved the vortex density profile with the lowest discrepancy values. Thus, the IDW4, IDW16, and bilinear interpolators will not be investigated more in this study.

5.7.5 Uniform flow around a cylinder

The uniform flow around the cylinder is implemented to validate the LB-overset method with the remaining interpolators. The inlet and outlet boundary conditions are the Zou and He velocity input and pressure output boundary conditions [85], while the lower and upper surfaces are straight velocity boundary conditions implemented according to the Inamuro boundary condition [87]. Moreover, a curved boundary treatment is implemented by applying the extrapolation Guo technique [30], which constructs the post-collision particle distribution function on the boundaries by extrapolating the density, velocity, and non-equilibrium distribution function from the neighboring fluid nodes. This section is divided into two main parts: uniform flow around a cylinder (non-rotating moving grid) and a rotating cylinder (rotating moving grid). The first section is implemented to validate the interpolators with an unsteady flow without a rotation effect. The latter is implemented to validate and compare the remaining interpolators with a more complex flow by adding the rotational effect.

5.7.5.1 Uniform flow a cylinder with a non-rotating grid

The simulation parameters for all interpolators are the fixed grid size of 1500×1500 nodes, Reynolds number $Re = 100$, reference length $L = D = N_y/32$, Mach number $Ma = u_{max}/c_s = 0.3$, and the fixed-to-moving grid size ratio of $N_y^f/N_y^m = 4$. Moreover, the kinematic viscosity ν is calculated from the Reynolds number $\nu = \frac{u_\infty L}{Re}$, consequently, the relaxation time $\tau = 3(\nu + 0.5)$. The simulation starts with zero velocity and unity density in the overall domain. The total forces acting on the boundary $\mathbf{F}_B$ are calculated according to the momentum exchange theory [32] as

$$\mathbf{F}_B = \sum_{k \in B_n} \sum_{i=1}^8 \xi_i (f_i^*(\mathbf{X}_k, t) + f_i^*(\mathbf{X}_k + \xi_i \delta_t, t)) \cdot \delta_k \quad (5.55)$$

Where B_n is the set of the boundary nodes indices, and δ_k equals 1 if the node is fluid and zero otherwise. The boundary nodes are the nodes that have both fluid and solid nodes in their neighbors. The post-collision particle distribution function at the boundary nodes $f_i^*(\mathbf{X}_b, t)$, and $f_i^*(\mathbf{X}_b + \xi_i \delta_t, t)$ is the post-collision particle distribution function of the neighboring nodes in the opposite direction. Moreover, the lift and drag coefficients in zero angle of attack are calculated from $\mathbf{F}_B = F_x \hat{i} + F_y \hat{j}$ as

$$C_L = \frac{2 F_y}{\rho u_\infty^2 D} \quad (5.56)$$

$$C_D = \frac{2 F_x}{\rho u_\infty^2 D} \quad (5.57)$$

Table 5.5 shows the lift coefficient amplitude C'_L , mean drag coefficient $\overline{C}_D$ and Strouhal number $St = f L/u_\infty$ for the remaining interpolators, where f is the vortex shedding frequency. The results show good agreement for the interpolators' metrics with the published data [90-93], solving a similar external flow. These results validate the overset method for unsteady flow and its integration with the interpolators.

Table 5.5. The lift and drag coefficients and Strouhal number for $Re=100$.

Interpolator	C'_L	$\overline{C}_D$	St
Present, Bicubic convolution	0.294	1.419	0.165
Present, Bicubic	0.303	1.41	0.164
Lima et al. [90]	0.29	1.39	0.164
Tseng and Ferziger [91]	0.29	1.42	0.164
Lai and Peskin [92]	0.3299	1.447	0.165
Berthelsen and Faltinsen [93]	0.34	1.38	0.169

5.7.5.2 Uniform flow around a rotating cylinder

In this section, two simulations are performed. The first has the fixed grid size of 1500×1500 , rotation ratio $a = 0.1$, Reynolds number $Re = 60$, Mach number $Ma = u_{max}/C_s = 0.17$, angular speed $\omega = u_\infty a/r = 4.1877E - 04$. The second simulation has the fixed grid size of 1200×1200 , rotation ratio $a = 0.5$, Reynolds number $Re = 20$, Mach number $Ma = u_{max}/C_s = 0.08$, angular speed $\omega = u_\infty a/r = 1.232E - 03$. The cylinder radius r , the reference length $L = D = N_x/32$, and the fixed-to-moving grid size ratio of $N_y^f/N_y^m = 4$. Moreover, the kinematic viscosity ν is calculated from the Reynolds number $\nu = \frac{u_\infty L}{Re}$, consequently, the relaxation time $\tau = 3(\nu + 0.5)$. The moving-to-fixed border is selected as a circle to avoid any border rapprochement or intersection.

The simulations start with zero velocity and unity density in the overall domain, with the rotation starting from the first time step. Then, the simulation continued for 100000 time-step, which covers about ten and twenty cycles of the moving grid around its center for the two cases, respectively. Figure 5.14 shows the velocity field for both simulations in the overlapped grids, illustrating the smooth transition between the grids even with high disturbances resulting from the rotation and the vortex street in the case of $Re=60$.

The total force acting on the boundary $\mathbf{F}_B$ are calculated as described in the previous section. However, this force should be transformed to be evaluated in the inertial frame $\mathbf{F}_B^i$ as :

$$F_B^i = \mathcal{R}^T F_B \quad (5.58)$$

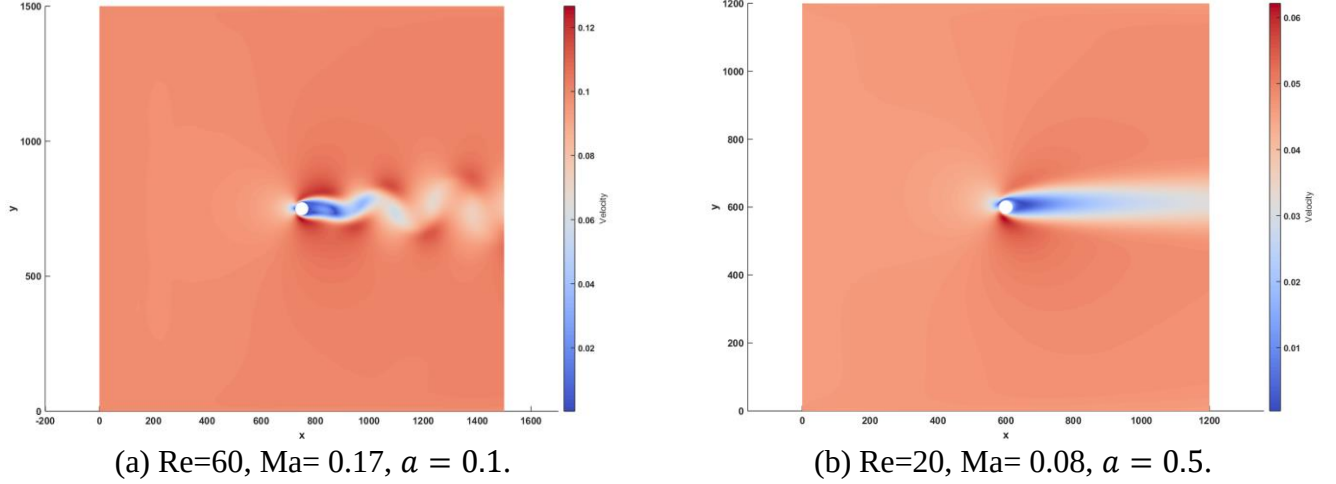


Figure 5.14. The velocity field is shown in the overlapped grid for the bicubic interpolator at t_f .

Table 4.5 shows the mean lift coefficient $\bar{C}_L$, amplitude lift coefficient C'_L , mean drag coefficient $\bar{C}_D$, and Strouhal number St for $Re = 60$ test case. All interpolators have almost the same results for all metrics, which agrees with the reference. Moreover, the differences in $\bar{C}_D$, which has a discrepancy of $\frac{1.418-1.379}{1.418} = 0.0275$, appears from the different methodologies and boundary conditions. For instance, Kang et al. solved the unsteady incompressible flow with boundary nodes conforming to the cylinder walls; on the other hand, this study uses an unsteady weakly-compressible solver, and the boundary nodes are not conforming to the cylinder wall. Despite that, the results for each metric have no marked difference from the reference. Moreover, Table 5.7 shows the lift and drag coefficients for $Re = 20$ test case, where the interpolators have good agreement with the references for each metric. These findings validate the LB-overset method with the bicubic and bicubic convolution interpolators and show that these interpolators have almost the same behavior for unsteady flow.

Table 5.6. The lift and drag coefficients and Strouhal number for rotation ratio = 0.1 and Re=60

Interpolator	$\overline{C}_L$	C'_L	$\overline{C}_D$	St
Present, Bicubic convolution	0.252	0.138	1.418	0.136
Present, Bicubic	0.252	0.138	1.417	0.136
Kang et al.[94]	0.250	0.135	1.379	0.136

Table 5.7. The lift and drag coefficients and Strouhal number for rotation ratio = 0.5 and Re=20.

Interpolator	C_L	C_D
Present, Bicubic convolution	1.390	2.057
Present, Bicubic	1.388	2.055
Badr et al. [95]	1.39	2.02
Kang et al.[94]	1.345	-
D. B. Ingham [105]	1.283	1.973
B. Fornberg [106]	-	2.0001

5.8 Conclusion

In the present study, the use of different interpolation functions in the LB-overset method was tested with a series of test cases. The objective consisted of comparing numerical accuracy and identifying the best interpolation function. We tested our numerical solver using steady and unsteady flows, with and without a solid object embedded in the overset grid. The results showed that the radial basis function interpolators show high pressure noise and discrepancies in the uniform flow, which could be explained using low interpolation node numbers regarding the selected shape parameter. On the other hand, increasing the nodes' number will significantly increase the computational resources, consequently decreasing the simulation performance.

The Poiseuille flow test case reveals that the nearest neighbor interpolator could not correctly implement the overset with the steady complex flow as it could not retrieve the correct velocity gradient. Moreover, the Taylor-Couette flow shows that the IDW4, IDW16, Bilinear, Bicubic convolution, and bicubic interpolators have the same accuracy for the steady flow field. Thus, bilinear, IDW4, IDW16, bicubic convolution, and bicubic are good choices for these steady flows.

The high-Re vortex advection test case shows that the IDW4, IDW16, and bilinear interpolators could not preserve the vortex density profile and had higher density discrepancies than the bicubic convolution and bicubic interpolators. On the other hand, the bicubic convolution and bicubic

interpolators preserved the density profile. Thus, the bicubic and bicubic convolution interpolators are recommended for the unsteady flow, with the bicubic interpolator being the most accurate.

The validation of the LB-overset method with the integrated bicubic and bicubic convolution algorithms is implemented by calculating the lift and drag coefficients around a rotating cylinder in a uniform flow, which is in good agreement with the references.

Acknowledgment

The author Abdallah ElSherbiny would like to thank Professor Jean-Yves Trepanier, URPEI, Department of Mechanical Engineering, Polytechnique Montréal, for constructive criticism of the manuscript and the fruitful discussions about the present study. Moreover, the authors acknowledge the support of the Natural Sciences and Engineering Council of Canada (NSERC), [funding reference number RGPIN-2020-04512].

5.9 APPENDIX A

The Appendix is divided into two sections; the first shows the Hermite polynomials and non-equilibrium high-order terms needed for implementing the equilibrium and non-equilibrium particle distribution function, while the second describes the interpolators' verification process.

5.9.1 Hermite polynomials and higher-order non-equilibrium distributions

The Hermite polynomials required for calculating equations (5.10) and (5.11) are presented in the equations from (5.59) to (5.61). For example, the term $\mathcal{H}_i^{(2)} : \mathbf{u}\mathbf{u}$ in equation (5.10) is calculated by equation (5.59) as $\mathcal{H}_i^{(2)} : \mathbf{u}\mathbf{u} = (\xi_{i,x}^2 - c_s^2)u_x^2 + (\xi_{i,y}^2 - c_s^2)u_y^2 + 2(\xi_{i,x}\xi_{i,y})u_xu_y$. Moreover, the high-order nonequilibrium terms are shown in equations (5.11) and (5.12) are illustrated in equations from (5.62) to (5.64).

$$\mathcal{H}_{i,xx}^{(2)} = \xi_{i,x}^2 - c_s^2, \quad \mathcal{H}_{i,xy}^{(2)} = \xi_{i,x}\xi_{i,y} \quad (5.59)$$

$$\mathcal{H}_{i,xyy}^{(3)} = (\xi_{i,x}^2 - c_s^2)\xi_{i,y} \quad (5.60)$$

$$\mathcal{H}_{i,xxyy}^{(4)} = (\xi_{i,x}^2 - c_s^2)(\xi_{i,y}^2 - c_s^2) \quad (5.61)$$

$$a_{1,xx}^{(2)} = \sum_i \mathcal{H}_{i,xx}^{(2)} (f_i - f_i^{eq}), \quad a_{1,xy}^{(2)} = \sum_i \mathcal{H}_{i,xy}^{(2)} (f_i - f_i^{eq}) \quad (5.62)$$

$$a_{1,xyy}^{(3)} = 2u_x a_{1,xy}^{(2)} + u_y a_{1,xx}^{(2)} \quad (5.63)$$

$$a_{1,xxyy}^{(4)} = 2(u_x a_{1,xyy}^{(3)} + u_y a_{1,xyy}^{(3)}) - (c_s^2 + u_x^2)a_{1,yy}^{(2)} - (c_s^2 + u_y^2)a_{1,xx}^{(2)} - 4u_xu_y a_{1,xy}^{(2)} \quad (5.64)$$

5.9.2 Interpolators verification

The interpolators' verification is implemented by defining a stencil of 16 nodes. This stencil includes 10^6 sample points located inside the central cell, as shown in Figure 5.15. These sample points are located on the centered cell to be implemented for the four and sixteen-node interpolators and unify the test inputs. Moreover, the stencil detection algorithm, described in section 5.6.1, uses a similar point location as it is located in the central cell. Then, a test function $\mu(x, y)$ is applied on the 16 nodes that precisely describes the interpolator equation. For example, the test function that describes the bilinear interpolation is $\mu(x, y) = 1 + x + y + xy$. Then, the variable quantities on the 16 nodes are calculated using the test function. Then, the interpolator interpolates the data from the 16 nodes to the 10^6 sample points. Finally, the values at the sample points and their L2 norm

$L_2 = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} (\phi_i - \mu_i)^2}$ are calculated. The interpolation function is verified when the L2 norm reaches machine accuracy for a test function of the same order.

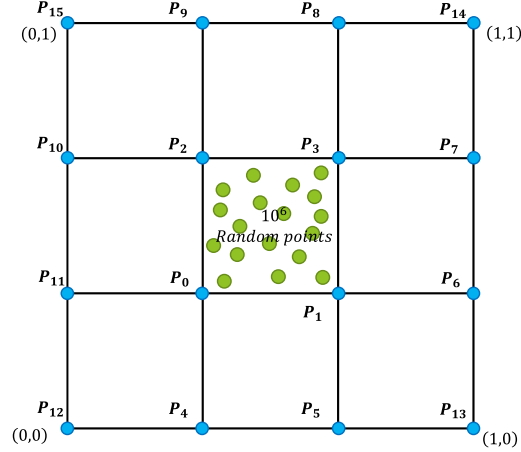


Figure 5.15. Schematic showing the stencil and the sample points for the interpolators' verification process.

Table 5.8 shows the different test functions used in the verification process of the interpolators. The nearest neighbor test function is a unity test function; thus, the nearest neighbor interpolator only has the machine's accuracy for this function. Moreover, the test function $\mu(x, y) = 1/\sqrt{(1 + R^2)}$ is used to verify the RBF interpolators, which only gives the machine's accuracy for this function. Otherwise, it adds errors even for the uniform function $\mu(x, y) = 1$. Furthermore, The IDW interpolator does not have a test function, as its interpolation is done through $\frac{\sum_i^N a_i \phi_i}{\sum_i^N a_i}$, and it has a machine accuracy for the unity test function.

The bilinear interpolator is also verified by retrieving the machine accuracy for the test function $\mu(x, y) = 1 + x + y + xy$, while it did not retrieve it for the nonlinear equation such as $\mu(x, y) = x - x^2$, because the bilinear interpolation equation does not include the terms x^2 or y^2 . On the other hand, the bicubic convolution has the machine accuracy of the test functions up to the quadratic equation $\mu(x, y) = 1 + x + y + xy + x^2 + y^2$ and it did not retrieve the machine's accuracy for $\mu(x, y) = \sum_{i=0}^3 \sum_{j=0}^3 x^i y^j$. Finally, the bicubic interpolator is verified by retrieving the machine's accuracy of the functions up to the third order $\mu(x, y) = \sum_{i=0}^3 \sum_{j=0}^3 x^i y^j$. Moreover, it has good results for other nonlinear functions, such as $\mu(x, y) = 1/\sqrt{(1 + R^2)}$.

Table 5.8. Interpolators verification table. The zero values in the tables represent the machine's accuracy.

[illegible]

CHAPTER 6 ARTICLE 2: Integration of Lattice Boltzmann-overset method with non-conforming quadtree mesh based on the combination of spatial and Lagrangian-link interpolated streaming technique

This chapter is a submitted manuscript that discusses the integration of the LB-overset method with the non-conforming quadtree mesh and introduces the spatial Lagrangian-link interpolated PDF streaming technique. The technical contribution of this chapter is the introduction of the developed spatial Lagrangian-link interpolated streaming technique, which allows the PDF streaming of the quadtree mesh based on 1D Lagrangian interpolation. This approach decreases the required computational resources and increases the simulation efficiency.

Abdallah ElSherbiny and Sébastien Leclaire, “Integration of Lattice Boltzmann-overset method with non-conforming quadtree mesh based on the combination of spatial and Lagrangian-link interpolated streaming technique,” submitted to *Journal of Computational Science* on Nov 7, 2023.

6.1 Abstract

This study integrates the Lattice Boltzmann overset approach with a non-conforming quadtree mesh to address fluid flow problems involving dynamic boundaries. The Lattice Boltzmann overset method employs two grids, one fixed and one movable, which can be computationally intensive due to the dual grid setup. A quadtree mesh is employed to reduce the number of nodes to mitigate this resource-demanding issue. Nonetheless, the use of the quadtree introduces challenges related to varying cell levels and spatial displacements. One of the approaches to address these challenges involves the use of an interpolated particle distribution function streaming technique. This study introduces an interpolation method, which initially applies spatial interpolation as a predictor step. Subsequently, this spatial predictor-interpolated value is utilized for a Lagrangian-link corrector interpolation. Furthermore, the study introduces a node-splitting technique aimed at enhancing the efficiency of the proposed interpolation scheme. The method’s order of accuracy is maintained without any degradation as a second order, and the flow around a rotating cylinder validates the method as the results align with previously published data.

Keywords: Lattice Boltzmann method, Overset method, Lagrangian-link interpolation, non-conforming quadtree mesh, moving boundary treatment.

6.2 Introduction

Solving the fluid flow with immersed moving bodies has long been a question of great interest with the lattice Boltzmann method (LBM) due to its wide application in industry. Several approaches solved this problem by performing moving boundary treatments, such as the moving overset method. The moving overset method was initially integrated with LBM as the sliding mesh interface approach [54, 82, 83], where the computational domain is divided into fixed and moving grids that interact through a shared interface ring. However, employing a single interface led to mass and momentum conservation challenges and generated numerical noise due to the relative motion [53]. Then, the LB-overset method was developed to solve this issue by introducing two communication borders: fixed-to-moving and moving-to-fixed borders [55, 56]. These communication borders act as boundary conditions [55]; the fixed-to-moving border acts as a boundary condition for the moving grid, and the moving-to-fixed border acts as a boundary condition for the fixed grid. The LB-overset method is a second-order accuracy method that defines the computational domain into two grids with their respective coordinate systems; these grids are the fixed grid and the moving grid, and the solid body is attached to the latter. Both grids share an overlapping region and communicate through two interfaces using interpolations.

The core of the LB-overset method is the reconstruction of PDFs on the communication borders, acting as boundary conditions for both grids. The PDFs' reconstruction is implemented by macroscopic interpolation [61] and microscopic interpolation, the so-called direct coupling method [56]. The macroscopic interpolation method interpolates the velocity, density, and flux tensor between the two grids on the borders. Then, the equilibrium and non-equilibrium PDFs are reconstructed from the interpolated macroscopic variables using LBM momentum equations. On the other hand, the microscopic interpolation performs the interpolation from the PDFs and then calculates the macroscopic variables from the interpolated PDFs to reconstruct the PDFs through their equilibrium and non-equilibrium parts. Moreover, it needs a rotation of the PDFs before the interpolation, which is a more complex process than the rotation of the macroscopic variables. Furthermore, it required interpolating the macroscopic stress tensor required to reconstruct the non-equilibrium PDFs.

Utilizing a uniform mesh in the LB-overset method or the standard LBM imposes significant limitations on its applicability for complex boundaries, which requires boundary treatments based

on grid refinement to increase the stability and accuracy of LBM and decrease the required computational resources. The boundary treatments based on grid refinement applied on LBM could be classified into body fitting and orthogonal mesh refinement methods.

First, the body-fitting methods modify the grid to conform to the boundaries, such as the body-conforming curvilinear mesh and the unstructured mesh methods. The body-conforming curvilinear mesh method [64, 65] gives accurate results but only applies to regular curvilinear boundaries such as circular ones. If the boundary is not near-circular, the mesh may be intersected or not uniform enough to adapt to the LBM. The LBM unstructured meshing methods, such as finite-difference LBM [33, 66, 67], finite volume LBM [35], and finite element LBM [68] are based on the concept of independent space and time discretization LBM [57]. Furthermore, an unstructured grid was introduced based on the flux solver LBM [7], which is based on interpolating macroscopic variables around the flux calculation point to construct the equilibrium particle distribution functions (PDFs) the flux solver needs. Besides confirming the boundaries, the unstructured mesh methods also perform grid refinement around the boundaries, giving accurate results as the unstructured mesh conforms to the non-regular boundaries. However, it adds more complex data structures, complex grid generation, and increased computational resources due to additional calculations, contradicting LBM's simplicity advantages on regular grids. Furthermore, some unstructured methods, such as finite-difference LBM, could produce numerical instability at low relaxation time LBM, requiring a high-order discretization scheme [33].

Second, the orthogonal grid refinement methods aim to refine the grid around the boundaries based on orthogonal cells, which could be classified into uniform and non-uniform orthogonal methods. The uniform orthogonal grid refinement methods, such as local grid refinement [39] and multi-block grid [25, 40], are based on performing the refinement with uniform refinement blocks.

The local grid refinement method introduces a grid refinement in locations of the most expected solution changes, which divides the mesh into two cells' blocks, fine and coarse cells. The fine-to-coarse displacement ratio n is not required to be two and can reach up to 50. As the grid size changes, the kinematic viscosity must be modified to retrieve the same viscosity and Reynolds number. Consequently, a relaxation parameter must be calculated in the fine mesh by knowing the coarse mesh relaxation parameter. However, the finer mesh performs the collision and streaming

steps n times to maintain the same time step in both blocks. Moreover, special treatments on the PDFs transferred between coarse and fine cells must be performed.

Similarly, the multi-block grid method generates fine and coarse blocks in the grid and calculates new relaxation parameters to maintain the same viscosity. Nevertheless, the coarse block displacement is doubled as the finer one [25], maintaining the same dynamic viscosity. Consequently, these blocks could be repeated to reach the required refinement and result in multi-blocks, significantly decreasing the required computational resources [25]. However, similar to the local grid refinement method, it requires special treatments on the blocks' boundaries, and the collision and streaming steps in each fine block are implemented two times its coarser one.

Then, the non-uniform orthogonal grid refinement methods, such as directional refinement method [107], non-uniform staggered cartesian grid (NSCG) [44], quadtree LBM [41, 42], and non-conforming orthogonal mesh [43] are based on performing the refinement with a non-uniform orthogonal mesh near the boundaries.

Cheng and Hung [107] introduced the directional refinement method, which applies refinements in specific directions, producing a non-uniform stretched grid containing rectangular cells with a uniform data structure. Moreover, they implemented the interpolated PDFs streaming in the non-uniform nodes using a spatial interpolation of nine surrounding nodes, which is performed using the multiplication of two 1-D Lagrange polynomials applied in the tangential and perpendicular directions.

Valero-Lara and Jansson [44] introduced the NSCG method, which presents a non-uniform refinement based on introducing rectangular cells and has four types of cells: coarse, fine, vertical-fine, and horizontal-fine, and implements scaling on the kinematic viscosity for each cell. Consequently, an arithmetic average relaxation parameter was presented and applied to corner nodes of the vertical-fine and horizontal-fine cells. This arithmetic average relaxation parameter allows all fluid points, whether coarse or fine, to be solved once each time step. However, it requires treatments based on the refinement cell in all lattice directions. Moreover, when applied to 1-D refinement, it reduces the nodes' number compared to local grid refinement. Conversely, it increases the necessary number of nodes when employed in 2D refinement, such as refinement around a cylinder.

The quadtree mesh is a data structure where the cells can be divided into four equivalent quadrants based on refinement requirements. These refinement requirements can be implemented by adding control points in the domain. If a cell contains a control point, it is split into four other cells till it reaches the maximum required level l_{max} . A cell with no splitting has a level equal to zero; with each split, the new cells have their level increased by one. The algorithm of the LB quadtree method in literature [41, 42] is based on the local grid refinement method, where the calculation is performed between each quadtree's levels l and the finest cells perform collision and streaming $2^{l_{max}}$ times the most coarse ones. This could lead to an extremely high processing time for high-level quadtree mesh. For instance, if the quadtree has $l_{max} = 6$, so the maximum level cells perform the collision and streaming $2^6 = 64$ times the low level cell. Moreover, the local grid refinement special treatment must be applied between each two consecutive level cells, adding more complexity and processing time.

Finally, Pellerin et al. [43] introduced the non-conforming orthogonal mesh method that generates a quadtree-like mesh with large aspect ratio cells, as the x and y direction splitting are separated. The core of the non-conforming LBM is using bicubic interpolated post-collision PDFs in the streaming step. These interpolated PDFs' locations are offset from the PDF pulling node with the smallest cell size dimensions, which leads to using the same time loop and relaxation time for all nodes in the non-conforming orthogonal mesh. However, it adds more complexity due to the bicubic interpolation, which is difficult and computationally costly in 2D and even more in 3D.

Regarding the grid refinement on the LB-overset method, only two studies perform grid refinement on the LB-overset method. Based on the standard LB-overset method, one study applies a multi-block refinement only on the fixed grid, while the moving grid is a uniform cartesian grid with the finest cells [61, 108]. The other one, based on the direct coupling LB-overset, applies the local grid refinement concept as the moving grid is a uniform cartesian grid with fine cells, while the fixed grid has coarse cells [56].

The present study aims to solve the flow around moving boundaries by integrating the LB-overset method with the non-conforming quadtree mesh based on the combination of spatial and lagrangian-link interpolated streaming technique. Our main goal of the study is to verify and validate this approach. The quadtree grid is selected to reduce the total number of nodes, consequently decreasing the computational resources, as the quadtree mesh always has a lower

number of nodes than the multi-block refinement for the same refinement level. Moreover, the interpolated PDF streaming technique is used instead of the methods used in the quadtree studies [41, 42] to avoid repeating the collision and streaming in the fine cells and the complex implementation on the different level boundaries.

The interpolated PDF streaming technique could be classified based on spatial (Eulerian) interpolation and link (Lagrangian) interpolation. The spatial interpolation methods [43, 107] can be adapted to the non-uniform grids. However, they are complex and require high computational resources and time due to the complex interpolation schemes. Moreover, the complexity and the computational resources will significantly increase for extending the methods to 3D. On the other hand, the Lagrangian-link interpolation scheme, such as Guo's boundary treatment [30], benefited from the LBM lagrangian nature and performed extrapolations on the non-equilibrium PDF on the link. However, it is not adapted to the non-uniform grids because the nodes are not aligned in the same link.

Thus, this study implements a combination of spatial and Lagrangian-link interpolation, which is straightforward, adapted to the non-uniform grids, and reduces the computational resources. Moreover, a streaming node splitting technique is developed to reduce the computational resources, splitting the streaming nodes into direct streaming, direct Lagrangian-link interpolation, and spatial Lagrangian-link interpolation nodes, which prevents the interpolation from being applied on all nodes. The direct streaming nodes are the most condensed area in the quadtree located in the cells with the highest level, where standard LBM streaming is applied and do not require interpolations. Moreover, the direct Lagrangian-link interpolation extends across much of the quadtree grid, affecting nodes surrounded by four square cells with spatial displacements identical and not equal to the minimum displacement. These nodes are situated in all level layers except the maximum level one. Furthermore, the spatial Lagrangian-link interpolation nodes are a few nodes located on the boundaries between the different levels.

The LB-overset method based on the recursive regularized BGK collision model is described in section 6.3. Then, the non-conforming quadtree mesh based on the spatial Lagrangian-link interpolated PDF streaming technique is discussed in section 6.4. Next, the results are discussed in section 6.5 by implementing four test cases: the uniform flow, Poiseuille flow, Taylor-Couette flow, and flow around a rotating cylinder to verify and validate the method. Finally, the conclusion

is articulated in section 6.6, and an appendix is attached at the end of the paper containing the equations related to the recursive regularized LBM.

6.3 The recursive regularized LB-overset method

This section briefly introduces the recursive regularized LBM and the LB-overset method. For more information, the reader can revise Latt and Chopard [60], Malaspina [62], and Yoo et al. [61] works. The recursive regularized LB-overset method performs the moving boundary treatments for solving flow around moving objects.

6.3.1 Recursive regularized LBM

The lattice Boltzmann method is an emerging matured CFD solver based on the principles of kinetic theory. It involves discretizing the lattice Boltzmann equation (LBE) in time and space, utilizing a discretized velocity space denoted as (ξ_i) . This discretization is achieved through Gauss-Hermite quadrature, enabling the evaluation of the PDF $f_i(\mathbf{x}, t)$ at a specific location, time, and lattice direction. The LBM's core operation revolves around two primary steps: collision and streaming.

Based on the Bhatnagar–Gross–Krook (BGK) model [96], the collision step is a relaxation process of the PDF $f_i(\mathbf{x}, t)$ towards the equilibrium PDF $f_i^{eq}(\mathbf{x}, t)$ after a specific time called relaxation time (τ). In other words, the collision step describes the collision of the PDF $f_i(\mathbf{x}, t)$ in a specific location, then redefining them with new values based on their equilibrium PDF $f_i^{eq}(\mathbf{x}, t)$. The streaming step represents the propagation of the post-collision PDFs to the surrounding nodes, each in its direction (i), which obtain the post-streaming PDFs on these nodes. In the following time step, the previous post-streaming PDFs are the current pre-collision ones. The collision and streaming can be represented as

$$f_i(\mathbf{x} + \xi_i \Delta t, t + \Delta t) - f_i(\mathbf{x}, t) = -\frac{\Delta t}{\tau} \left(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t) \right) \quad (6.1)$$

The equilibrium PDF f_i^{eq} is typically defined using the Maxwell-Boltzmann distribution, and its formula change depends on the lattice model. For instance, the D2Q9 model is applied for two dimensions flows with population discretized into nine directions ($i = 0:8$). Moreover, the equilibrium PDF is calculated using the weighting function ω_i and the local microscopic velocity

(ξ_i) associated with the lattice direction i , and the macroscopic variables presented in the flow density (ρ) and velocity ($\mathbf{u}$) as

$$f_i^{eq}(\mathbf{x}, t) = w_i \rho \left(1 + \frac{\xi_i \cdot \mathbf{u}}{c_s^2} + \frac{1}{2 c_s^4} \mathcal{H}_i : \mathbf{u} \mathbf{u} \right) \quad (6.2)$$

Where the Hermite polynomials $\mathcal{H}_{i\alpha\beta} = \xi_{i\alpha}\xi_{i\beta} - c_s^2 \delta_{\alpha\beta}$, the lattice weights is w_i , the lattice speed of sound is c_s , and $\delta_{\alpha\beta}$ is the Kronecker delta. Moreover, the flow density (ρ) and velocity ($\mathbf{u}$) are retrieved from the PDF zeroth and first moments for the D2Q9 model as

$$\rho = \sum_{i=0}^8 f_i \quad (6.3)$$

$$\rho \mathbf{u} = \sum_{i=0}^8 \xi_i f_i \quad (6.4)$$

The regularized BGK model is based on maintaining the second-order Knudsen number ϵ of the Chapman–Enskog expansion [97], which allows the definition of the regularized PDF f_i^{Reg} based on the second-order non-equilibrium PDF f_i^{Neq} as

$$f_i^{Reg} = f_i^{Neq} + f_i^{eq} \quad (6.5)$$

The second-order non-equilibrium PDF f_i^{Neq} is reconstructed using the non-equilibrium momentum flux tensor $\mathbf{\Pi}^{Neq}$ [60] as

$$f_i^{Neq} \approx f_i^{(1)} = \frac{w_i}{2 c_s^4} \mathcal{H}_i \cdot \mathbf{\Pi}^{Neq} \quad (6.6)$$

The non-equilibrium momentum flux tensor $\mathbf{\Pi}^{Neq}$ is calculated through the momentum flux tensor $\mathbf{\Pi}$ and its equilibrium part $\mathbf{\Pi}^{eq}$ as

$$\mathbf{\Pi}^{Neq} = \mathbf{\Pi} - \mathbf{\Pi}^{eq} \quad (6.7)$$

$$\Pi_{\alpha\beta} = \sum_{i=0}^8 \xi_{i\alpha} \xi_{i\beta} f_i \quad (6.8)$$

$$\Pi_{\alpha\beta}^{eq} = \sum_{i=0}^8 \xi_{i\alpha} \xi_{i\beta} f_i^{eq} \quad (6.9)$$

Where α, β covers all possible combinations of the two directions x, y as xx, xy, yx, yy . The regularized LBM significantly improves stability and accuracy by truncating only up to the second-order non-equilibrium part. This approach is computationally efficient, requiring only a few additional steps to calculate the momentum flux tensor and the non-equilibrium distribution function [60].

In contrast to the regularized LBM, which utilizes second-order terms in the equilibrium and non-equilibrium PDFs, the recursive regularized LBM incorporates higher-order terms. This enhancement results in solutions that are both more stable and accurate [62, 63]. Furthermore, the higher-order non-equilibrium distribution functions are derived from a recursive formula based on the second-order non-equilibrium tensor $\mathbf{a}_1^{(2)}$, resulting in a velocity space discretization error of approximately order $\mathcal{O}(Ma^{n+1})$ [62], which effectively reduces the compressibility error that arises due to the Mach number Ma . The equilibrium and non-equilibrium PDFs up to fourth-order are shown in equations (5.10) and (5.11).

$$\mathbf{a}_1^{(2)} = \mathbf{\Pi}^{Neq} = \sum_{i=0}^8 \mathcal{H}_i^{(2)} (f_i - f_i^{eq}) \quad (6.10)$$

$$f_i^{eq} = w_i \rho \left(1 + \frac{\boldsymbol{\xi}_i \cdot \mathbf{u}}{c_s^2} + \frac{1}{2 c_s^4} \mathcal{H}_i^{(2)} : \mathbf{u}\mathbf{u} + \frac{1}{2 c_s^6} \left(\mathcal{H}_{ixxy}^{(3)} u_x^2 u_y + \mathcal{H}_{ixyy}^{(3)} u_y^2 u_x \right) + \frac{1}{4 c_s^8} \mathcal{H}_{ixxyy}^{(4)} u_x^2 u_y^2 \right) \quad (6.11)$$

$$f_i^{Neq} = w_i \left(\frac{1}{2 c_s^4} \mathcal{H}_i^{(2)} : \mathbf{a}_1^{(2)} + \frac{1}{2 c_s^6} \left(\mathcal{H}_{ixxy}^{(3)} a_{1, xxy}^{(3)} + \mathcal{H}_{ixyy}^{(3)} a_{1, xyy}^{(3)} \right) + \frac{1}{4 c_s^8} \mathcal{H}_{ixxyy}^{(4)} a_{1, xxyy}^{(4)} \right) \quad (6.12)$$

6.3.2 LB-overset method

The LB-overset method presented in this section is based on the work of Yoo et al. [61]. The LB-overset approach involves employing two grids with distinct reference frames. The fixed grid adopts the inertial frame, whereas the moving boundary is affixed to the moving grid. Moreover, the fixed grid is addressed through the LBE in equation (4.1). In contrast, the LBE for the moving grid incorporates an extra discrete forcing term, denoted as $F_i(\mathbf{x}, t)$ or the Guo forcing term [21] as shown in equation (5.13).

$$f_i(\mathbf{x} + \xi_i \Delta t, t + \Delta t) - f_i(\mathbf{x}, t) = -\frac{\Delta t}{\tau} \left(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t) \right) + \left(1 - 0.5 \frac{\Delta t}{\tau} \right) F_i(\mathbf{x}, t) \quad (6.13)$$

The discrete force term $F_i(\mathbf{x}, t)$ introduces the influence of the inertial density force $\mathbf{F}(\mathbf{x}, t)$ derived from relative motion. This inertial force comprises the linear acceleration $-\dot{\mathbf{V}}$, centripetal acceleration $-2\boldsymbol{\omega} \times \mathbf{u}^c$, Coriolis acceleration $-\mathbf{u}^c \times \boldsymbol{\omega}$, and Euler acceleration $-\dot{\boldsymbol{\omega}} \times \mathbf{r}$, as shown in equations (5.15) and (5.16). Moreover, calculating the discrete force term and the inertial density force are based on the corrected velocity field $\mathbf{u}^c$ to retrieve a second-order accuracy. The corrected velocity is calculated using the half-force correction [56] shown in equation (6.16).

$$F_i(\mathbf{x}, t) = w_i \left(\frac{(\xi_i - \mathbf{u}^c) \cdot \mathbf{F}}{c_s^2} + \frac{(\xi_i \cdot \mathbf{u}^c) (\xi_i \cdot \mathbf{F})}{c_s^4} \right) \quad (6.14)$$

$$\mathbf{F} = -\rho(\dot{\mathbf{V}} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}) + 2\boldsymbol{\omega} \times \mathbf{u}^c + \dot{\boldsymbol{\omega}} \times \mathbf{r}) \quad (6.15)$$

$$\rho \mathbf{u}^c(\mathbf{x}, t) = \sum_i \xi_i f_i(\mathbf{x}, t) + \mathbf{F}(\mathbf{x}, t) \Delta t / 2 \quad (6.16)$$

The corrected velocity is applied to calculate the equilibrium PDF $f_i^{eq}(\rho, \mathbf{u}^c)$ in equations (4.4) and (5.10), which leads to an implicit problem as it is also a function of the inertial density force $\mathbf{F}(\mathbf{x}, t)$. This issue can be resolved by applying the accurate solution for the Coriolis acceleration $\frac{d\mathbf{u}}{dt} = -2\boldsymbol{\omega} \times \mathbf{u}$ [99], as:

$$\mathbf{u}^c = \begin{bmatrix} u_x \\ u_y \end{bmatrix}_{t+\frac{\Delta t}{2}} = \begin{bmatrix} \cos(\omega \Delta t / 2) & \sin(\omega \Delta t / 2) \\ -\sin(\omega \Delta t / 2) & \cos(\omega \Delta t / 2) \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}_t \quad (6.17)$$

The fixed and moving grid communicates the macroscopic variables such as density, velocity, and flux tensor through two communication borders: fixed-to-moving and moving-to-fixed. These boundaries serve as boundary conditions for both grids, where the PDFs $f_i(\mathbf{x}, t)$ are reconstructed. This reconstruction relies on interpolating the macroscopic variables at each node along these boundaries, using data from the neighboring nodes of the other grid.

The fixed-to-moving treatment seeks to reconstruct the PDFs at the boundaries of the moving grid. Initially, on the fixed grid, the macroscopic variables $(\rho, \mathbf{u}, \boldsymbol{\Pi}^{Neq})$ are interpolated onto the projected borders of the moving grid. This interpolation process utilizes the neighboring nodes for implementation, using equations (5.19) to (5.21), where the interpolated function is ϕ , and the superscript (f, m) indicates the variable in the fixed and moving grids, respectively. Subsequently,

the geometric adjustment of the velocity and flux tensor is applied to accommodate the disparity in coordinate systems, utilizing the rotation matrix $\mathcal{R}$, as articulated in equations (5.22) to (6.23).

$$\rho^f(\mathbf{x}^m) = \phi\left(\rho^f(\mathbf{x}^f)\right) \quad (6.18)$$

$$\mathbf{u}^f(\mathbf{x}^m) = \phi\left(\mathbf{u}^f(\mathbf{x}^f)\right) \quad (6.19)$$

$$\mathbf{\Pi}^{Neq,f}(\mathbf{x}^m) = \phi\left(\mathbf{\Pi}^{Neq,f}(\mathbf{x}^f)\right) \quad (6.20)$$

$$\mathbf{u}_R^f(\mathbf{x}^m) = \mathcal{R} \cdot \mathbf{u}^f(\mathbf{x}^m) \quad (6.21)$$

$$\mathbf{\Pi}_R^{Neq,f}(\mathbf{x}^m) = \mathcal{R} \cdot \mathbf{\Pi}^{Neq,f}(\mathbf{x}^m) \cdot \mathcal{R}^T \quad (6.22)$$

$$\mathcal{R} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \quad (6.23)$$

Next, the velocity is adjusted by incorporating the influence of non-inertial acceleration through the subtraction of the relative velocity, expressed as $\mathbf{u}_{R,A}^f(\mathbf{x}^m) = \mathbf{u}_R^f(\mathbf{x}^m) - \mathbf{v}_r$, where $\mathbf{v}_r = \boldsymbol{\omega} \times \mathbf{r}$. Here, the subscripts (R, A) denote the adjustments due to the geometric rotation and non-inertial acceleration, respectively. Moreover, the non-equilibrium momentum tensor remains unchanged, as the grid rotation does not affect the shear stresses [61]. Following this, the macroscopic variables at the moving grid border $\mathbf{x}^m$ are redefined based on the modified values from the fixed grid, as $\mathbf{u}^m(\mathbf{x}^m) = \mathbf{u}_{R,A}^f(\mathbf{x}^m)$, $\mathbf{\Pi}^{Neq,m}(\mathbf{x}^m) = \mathbf{\Pi}^{Neq,f}(\mathbf{x}^m)$, and $\rho^m(\mathbf{x}^m) = \rho^f(\mathbf{x}^m)$. Finally, the PDFs at the border f_i^m are reconstructed by employing the redefined macroscopic variables at the borders. This reconstruction involves reconstituting both equilibrium and non-equilibrium PDFs using equations (5.10) and (5.11), along with the incorporation of the half-force correction force term F_i in equation (5.15) as:

$$f_i^m(\mathbf{x}^m, t) = f_i^{eq}(\rho^m(\mathbf{x}^m), \mathbf{u}^m(\mathbf{x}^m)) - 0.5 F_i(\rho^m(\mathbf{x}^m), \mathbf{u}^m(\mathbf{x}^m)) + f_i^{Neq}(\mathbf{u}^m(\mathbf{x}^m), \mathbf{\Pi}^{Neq,m}(\mathbf{x}^m)) \quad (6.24)$$

The moving-to-fixed treatment is carried out following the same procedures as the fixed-to-moving one, with the notable difference of adding the relative velocity to the total velocity, expressed as $\mathbf{u}_A^m(\mathbf{x}^f) = \mathbf{u}^m(\mathbf{x}^f) + \mathbf{v}_r$, instead of subtracting it. This adjustment is made before the interpolation and the grid rotation [61]. Additionally, there is no need for the half-force

correction when reconstructing the PDFs on this border, as the fixed frame serves as the inertial reference frame [56].

6.4 Non-conforming quadtree mesh based on interpolated PDF streaming technique

In this section, the quadtree mesh generation is described in section 6.4.1, with the illustration of the streaming nodes splitting. Then, the spatial Lagrangian-link interpolated PDF streaming technique is discussed in section 6.4.2. Finally, section 6.4.3 articulated the spatial Lagrangian-link technique.

6.4.1 Quadtree mesh and streaming nodes' splitting

The non-conforming quadtree mesh is a refinement mesh technique where the mesh nodes do not necessarily lay on the boundaries. It also reduces the total number of nodes, reducing the required computational resources. The quadtree mesh generation core is splitting the cells into four equivalent quadrants by specified refinement criteria. This refinement criteria can be implemented by defining control points in the domain. Each cell that contains a control point is divided into four new cells: northeast NE, northwest NW, southeast SE, and southwest SW, and five new nodes located on half of the spatial spacing, as shown in Figure 6.1. Moreover, the newly generated cell's level increased by one, and the refinements continued until they reached the maximum required level.

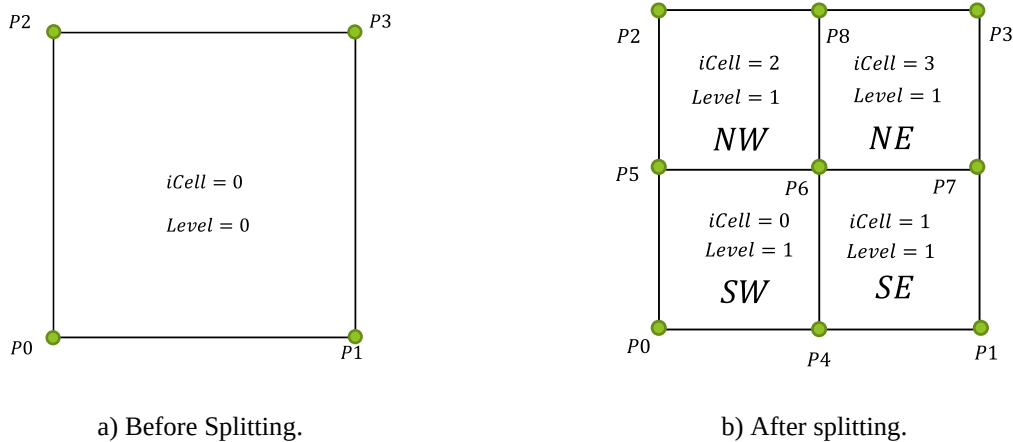


Figure 6.1 Schematic shows the quadtree mesh splitting.

Furthermore, a boundary node detection algorithm has been created to categorize the grid into solid, boundary, and fluid nodes using the winding number algorithm [88], as shown in Figure 6.2.

This classification allows us to conduct collision and streaming operations for the fluid nodes, implement suitable boundary treatments for the boundary nodes, and effectively exclude any solid nodes from the simulations, reducing the required computational resources.

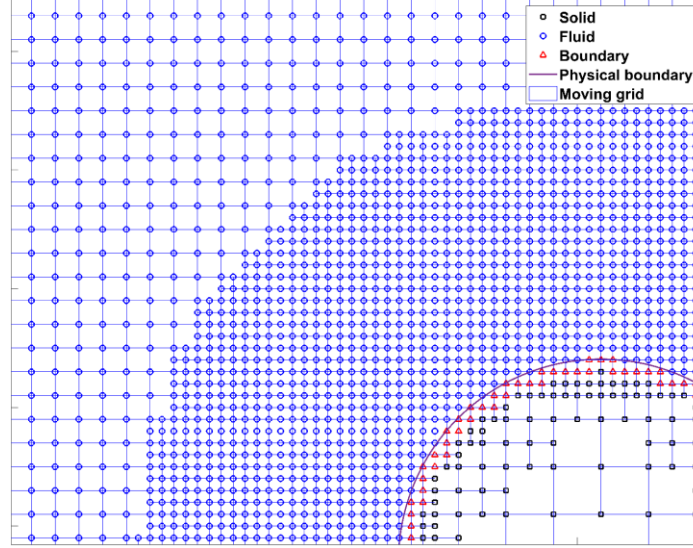


Figure 6.2 An example of a circular boundary shows the grid nodes' classification as solid, fluid, and boundary nodes. The solid node is the unfilled black circle, the fluid node is the unfilled blue diamonds, the boundary nodes are the unfilled red triangles, and the physical boundary is the continuous purple line.

As discussed in the introduction, the spatial Lagrangian-link interpolation is not required to be implemented on all nodes, and it is implemented only on the nodes that are not located on the link. Thus, the quadtree domain is further classified into three node categories: direct streaming, direct Lagrangian-link, and spatial Lagrangian-link nodes, as shown in Figure 6.3.

The "direct streaming nodes" are nodes where all surrounding cells share the same minimal spatial displacement within the domain. As a result, these nodes possess the accurate local relaxation time and viscosity, obviating the need for any interpolation. Thus, they directly undergo a streaming step similar to the standard LBM. Furthermore, these direct streaming nodes are concentrated in the domain's most compact region, eliminating the interpolation requirement and reducing computational effort. These nodes can be identified by employing a condition based on the cells surrounding them, which share the same level, and this level is the highest grid level.

Then, the "direct Lagrangian-link nodes" are the majority of the quadtree mesh. These nodes employ a straightforward Lagrangian-link interpolation method, avoiding the necessity to perform spatial interpolation on all nodes, resulting in significant reductions in computational resources and

effort. These nodes can be identified by employing a condition based on the cells surrounding them, which share the same level, and this level does not equal the highest grid level.

Finally, the "spatial Lagrangian-link nodes" represent the nodes where the spatial Lagrangian-link interpolation is carried out where the spatial displacements of the surrounding cells are not equal. These nodes are exclusively present along the boundaries that separate different levels of layers, constituting a minority among all grid nodes. Consequently, the need for spatial Lagrangian-link interpolation arises only for a few nodes, resulting in significant computational resource savings. These nodes can be identified by employing a condition based on the cells surrounding them, which do not have the same level. Furthermore, this condition could be further enhanced by considering the link direction and removing the collinear nodes; however, it is not considered in this study because the quantity of such nodes is relatively limited, which almost does not affect the computational resources. More details are provided about the Lagrangian-link interpolation in section 6.4.2 and the spatial Lagrangian-link interpolation in section 6.4.3.

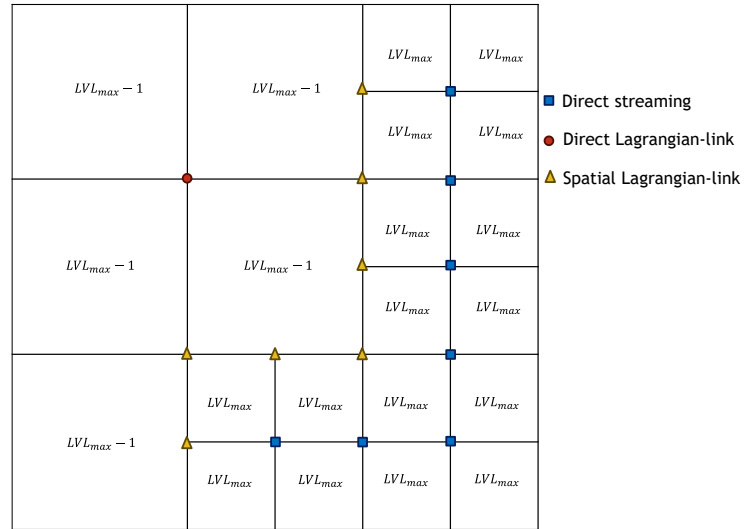


Figure 6.3 Schematic shows the three classifications of the streaming nodes. The blue squares are the direct streaming, the green circles are the direct Lagrangian-link, and the yellow triangles are the spatial Lagrangian-link nodes.

6.4.2 Lagrangian-link interpolated PDF streaming technique

The interpolated PDF streaming is based on interpolating the post-collision PDFs f_i^* on specific locations, which enables the same relaxation time and time step for all nodes in the grid despite their levels. These interpolated PDFs' locations P_{interp} are offset from the pulling node with the smallest cell size in the quadtree mesh, as shown in Figure 6.4. As LBM also has Lagrangian bases,

interpolating the post-collision PDFs can be implemented on the Lagrangian path with the particles moving on the same link. Consequently, the interpolated post-collision PDFs are obtained using the post-collision PDFs of the nodes on the link, revealing the direct lagrangian-link interpolation. This interpolation is a straightforward 1-D interpolation that consumes low computational resources compared to spatial interpolators such as bicubic interpolator. In this study, the interpolation is performed using the Lagrangian polynomial interpolation. This interpolation is applied in all directions, resulting in all required post-streaming interpolated PDFs used in the streaming process. The interpolated PDF $f_i^*(P_{interp})$ can be reconstructed using the three points A , B , and C which have enough information $f_i^*(A)$, $f_i^*(B)$, and $f_i^*(C)$ to reconstruct a quadratic parabola as:

$$f_i^*(P_{interp}) = \phi_{Lag}(A, B, C, f_i^*(A), f_i^*(B), f_i^*(C)) \quad (6.25)$$

Where ϕ_{Lag} is the 1-D quadratic Lagrangian interpolation function.

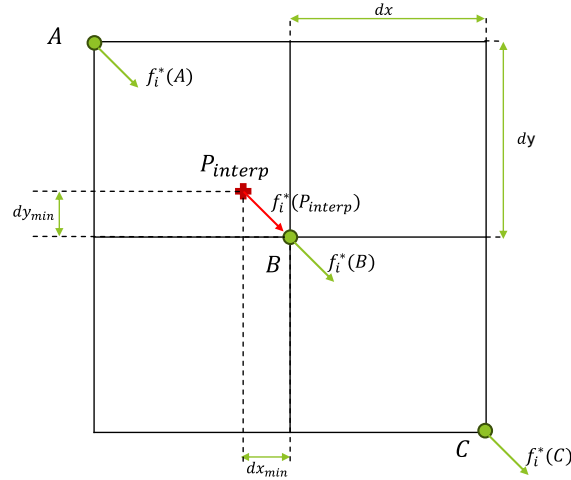


Figure 6.4 The interpolated post-collision PDFs. The green circles and arrows are the lagrangian-link nodes and their corresponding post-collision PDFs, and the red cross and arrow are the interpolation location and the interpolated post-collision PDF.

6.4.3 Spatial Lagrangian-link interpolated PDF streaming technique

The quadtree mesh is a non-uniform grid, which does not guarantee the correct lagrangian path of the particles as the nodes may not be on the same link, as shown in Figure 6.5.a). The node D is not collinear with $\overline{HI}$, so if we used directly the location of the node D for interpolation it will lead to numerical instability. Thus, the spatial Lagrangian-link interpolated scheme is introduced to implement at first a predictor step where the post-collision PDF is interpolated on the path's known

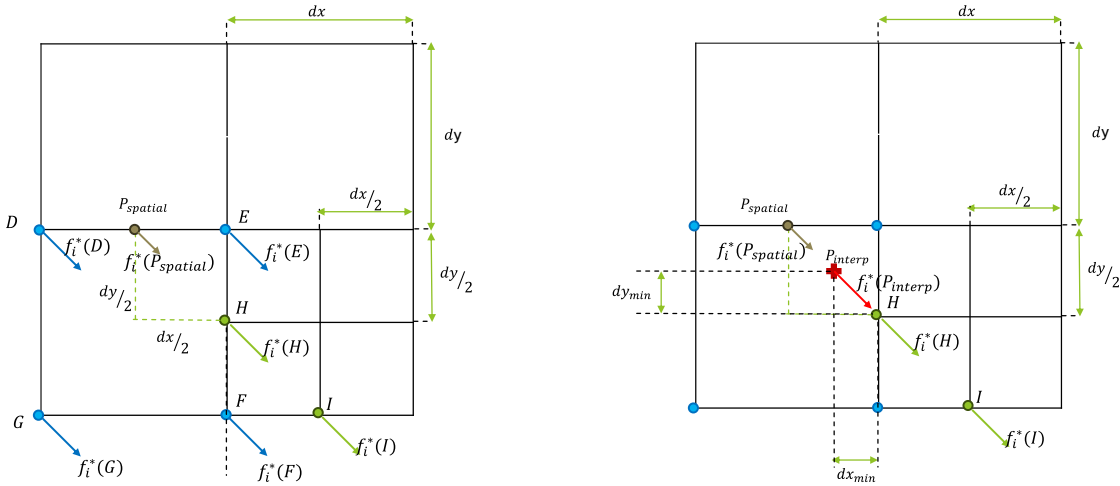
location $P_{spatial}$ using a spatial interpolator, as shown in Figure 6.5.a). The spatial interpolation implemented in this study is the bilinear interpolation, which uses the four nodes D , E , F , and G and their corresponding post-collision PDFs $f_i^*(D)$, $f_i^*(E)$, $f_i^*(F)$, and $f_i^*(G)$ as:

$$f_i^*(P_{spatial}) = \phi_{Spatial}(D, E, F, G, f_i^*(D), f_i^*(E), f_i^*(F), f_i^*(G)) \quad (6.26)$$

Then, the lagrangian-link interpolation is implemented as a corrector step to calculate $f_i^*(P_{interp})$ using the predicted interpolated post-collision PDF at $P_{spatial}$ and the other two existing nodes on the link H , and I , as shown in Figure 6.5.b), using the following equation:

$$f_i^*(P_{interp}) = \phi_{Lag}(P_{spatial}, H, I, f_i^*(P_{spatial}), f_i^*(H), f_i^*(I)) \quad (6.27)$$

Where the $\phi_{Spatial}$ is a bilinear interpolator acting as a predictor, while the ϕ_{Lag} is a quadratic 1-D interpolator acting as a corrector.



a) Schematic of the predictor step of spatial interpolation as node D are not colinear with $\overline{HI}$.

b) Schematic of the second step of the spatial Lagrangian-link interpolation, showing the Lagrangian-link interpolation using the predicted spatial PDF.

Figure 6.5 Schematics showing the spatial Lagrangian-link interpolation. The blue circles and arrows are the surrounding nodes and PDFs used in spatial interpolation, the grey circle is the location of the spatial interpolation location, and the red cross is the interpolated post-collision PDF location.

This approach significantly minimizes the computational resources required compared to utilizing a purely spatial interpolated streaming scheme like the bicubic method introduced by Pellerin et al. [43]. This reduction is achieved by diminishing the required nodes needed for interpolation. Moreover, it is a straightforward process that does not necessitate extensive

computational effort, primarily because the spatial Lagrangian-link interpolation involves explicit equations. Moreover, it is applicable to any orthogonal type of grid, including grids with rectangular cells, and is more suitable for 3D extension due to lower computational resources and effort. Furthermore, it is important to note that the spatial interpolation is not entirely quadratic, as it utilizes bilinear interpolation. However, it is employed as a predictor step on a very small number of nodes, which is expected to have a negligible impact on the solution's accuracy. Additionally, using bilinear requires only four points and is good for parallelization as it keeps the scheme as local as possible.

6.5 The Result and Discussion

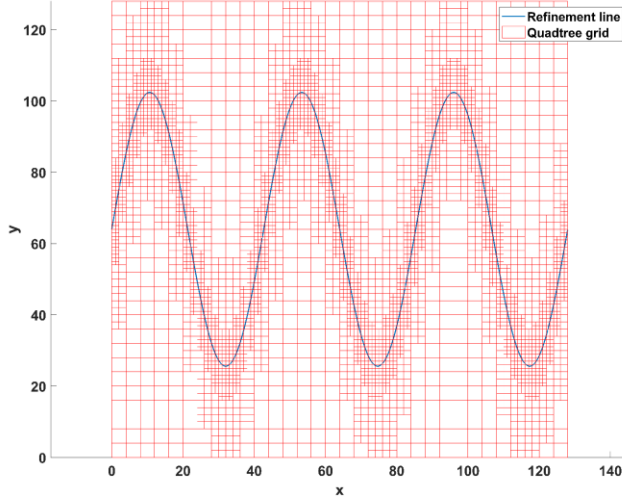
This section aims to verify and validate the LB-overset with spatial Lagrangian-link interpolated PDF streaming method by discussing four test cases. In section 6.5.1, the uniform flow is implemented to ensure no slight flow distortion due to the moving grid motion and the streaming interpolated method applied in both the fixed and moving grid. Then, section 6.5.2 discussed the Poiseuille flow, which presents a more complex flow as the velocity gradient is introduced. Moreover, the second order of accuracy is verified for the LB-overset with spatial Lagrangian-link interpolated PDF streaming method. After that, the Taylor-Couette flow is illustrated in section 6.5.3, where the method is applied to a curved boundary problem and verified by retrieving the analytical solution velocity profile. Finally, the flow around a rotating cylinder is implemented at different Reynolds numbers and rotational ratios to validate the present study in section 6.5.4.

For all the test cases, the moving grid is the quadtree grid, while the fixed grid is a uniform grid with displacement equal to the largest moving grid's displacement. Moreover, All the test cases implemented with the lattice system of units, where the minimum spatial displacement, time step, and density are unity $\Delta x_{min} = \Delta y_{min} = \Delta t = \rho = 1$; thus, the fixed grid spatial displacement is multiplied by $2^{LVL_{max}}$ as $\Delta x_f = \Delta y_f = 2^{LVL_{max}}$.

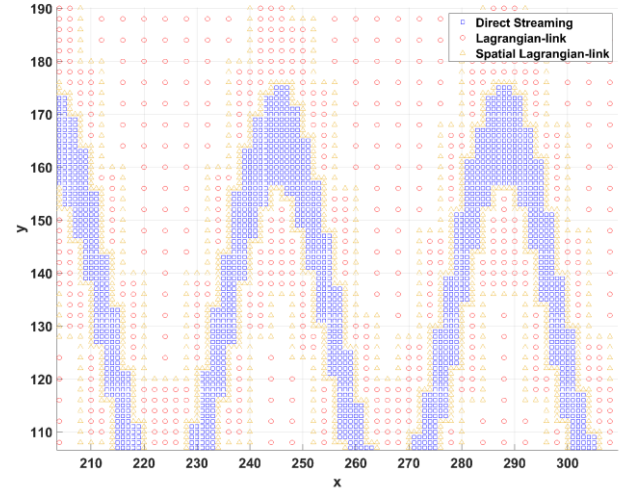
6.5.1 Uniform flow

Uniform flow is employed to ensure no flow field distortion, even in the presence of a rotating grid and the utilization of a quadtree mesh. The quadtree mesh is designed with a sinusoidal wave to apply a non-uniform pattern on the moving grid, as shown in Figure 6.6.a). Moreover, the moving grid nodes are separated into three nodes of streaming: direct streaming, direct Lagrangian-

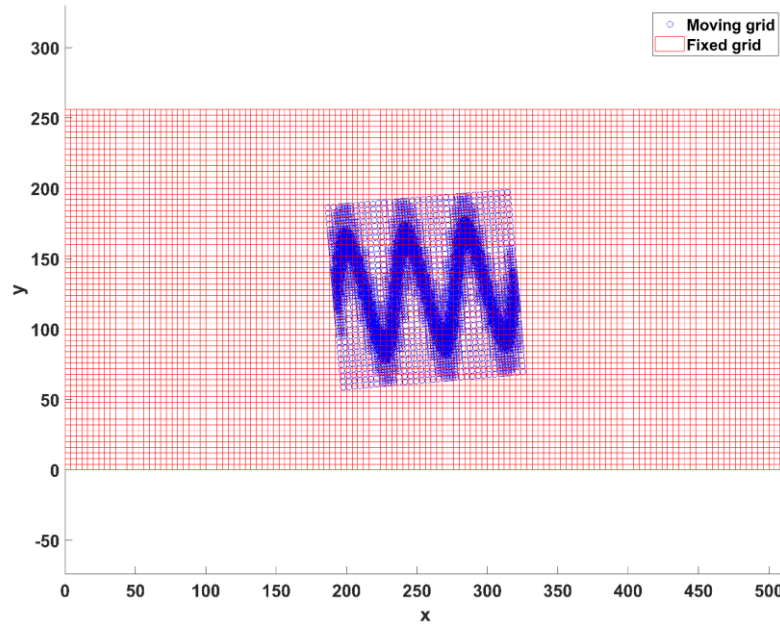
link, and spatial Lagrangian-link nodes, as shown in Figure 6.6.b). Furthermore, the overlapped fixed and moving grids are shown in Figure 6.6.c), where the moving grid is based on sinusoidal wave refinement, and the fixed grid is scaled with $2^{LVL_{max}}$ to maintain the minimum spatial displacements equals one.



a) Quadtree grid refinement.



b) Streaming nodes splitting of the moving grid. The blue squares are the direct streaming nodes, the red circles are the direct Lagrangian-link nodes, and the orange triangles are the spatial Lagrangian-link nodes.



c) The overlapped fixed and moving grids.

Figure 6.6 The mesh refinement based on sinusoidal wave control points.

The inlet and outlet boundary conditions are Zou and He velocity inlet and pressure outlet [85]. The upper and lower boundary conditions are straight velocity implemented according to the

Inamuro boundary condition [87]. The simulation parameters are the fixed grid size of 128×64 cells, relaxation time $\tau = 0.7$, Reynolds number $Re = \frac{u_\infty L}{\nu} = 100$, reference length $L = N_y^f = 64$, kinematic viscosity $\nu = \frac{1}{3}(\tau - 0.5)$, Mach number $Ma = u_{max}/c_s = 0.1776$, rotational speed $\omega = 10^{-4}$, and the fixed-to-moving grid size ratio of $N_y^f/N_y^m = 2$.

The simulation starts with zero velocity and unity density in the overall domain, with the moving grid starting rotation from the first time step t_s . The error L_1^u is calculated as

$$L_1^u = \frac{1}{N} \sum_{k=0}^{N-1} \frac{\|\mathbf{u}(\mathbf{x}_k) - \mathbf{u}_{ref}(\mathbf{x}_k)\|}{\max(\mathbf{u}_{ref})} \quad (6.28)$$

Where the reference velocity $\mathbf{u}_{ref}$ is the velocity of the uniform flow field, the nodes' index is k , and N is the total number of nodes. The moving grid rotates for $t_f = 100000$ time-step, which covers more than two rotations of the moving grid around its center.

The simulation is also tested regarding the error L_1^u , its time-average value $\hat{L}_1^u$, the pressure noise at the final time $\left. \frac{\partial P}{\partial t} \right|_{t_f}$, and its space-average value $\frac{\partial \bar{P}}{\partial t}$ as

$$\hat{L}_1^u = \frac{1}{(t_f - t_s)} \sum_{t=t_s}^{t_f} L_1^u(t) \quad (6.29)$$

$$\left. \frac{\partial P}{\partial t} \right|_{t_f} = \frac{P_{t_f} - P_{t_f - \Delta t}}{\Delta t} \quad (6.30)$$

$$\frac{\partial \bar{P}}{\partial t} = \frac{1}{N} \sum_{k=0}^{N-1} \left. \frac{\partial P}{\partial t} \right|_{t_f, k} \quad (6.31)$$

The error L_1^u is calculated to evaluate the effect of the rotation and the streaming interpolation technique on a non-uniform grid refinement. The average-time value indicates the performance over time after the grid motion. The pressure noise indicates the noise accompanied by the particle distribution function reconstruction on the border during the rotation. Moreover, the particle distribution function directly affects the pressure calculation on each node. Thus, the pressure change with time indicates the accuracy of PDF reconstruction and the streaming interpolation effect.

Table 5.1 shows that the values of L_1^u , $\hat{L}_1^u$, and $\frac{\partial \bar{P}}{\partial t}$ at the final time step t_f , which reveal that the error is almost near the machine's accuracy and the simulation is not affected by the spatial Lagrangian-link streaming technique and the rotation of the moving grid. This is expected since the flow is constant and the 1D quadratic interpolation and the 2D bilinear interpolation are used, so they should capture the constant profile accurately.

Table 6.1. The L_1 norm error and pressure noise after grid rotation at t_f .

	L_1^u	$\hat{L}_1^u$	$\frac{\partial \bar{P}}{\partial t}$
Present study	1.4687e-09	1.1572e-09	-3.3019e-14

6.5.2 Poiseuille flow

Poiseuille flow is introduced to add flow complexity by including a velocity gradient, testing the impact of the presented method on the simulation, and testing the order of accuracy as the flow solution is parabolic in the continuum limit. The moving quadtree grid is based on a sinusoidal control points wave similar to Figure 6.6. The inlet and outlet boundary conditions are velocity inlet and pressure outlet evaluated using the Zou and He [85], while the far field of the lower and upper surfaces are halfway bounceback [28].

The simulation is performed for four sizes of the fixed grid as 32×16 , 64×32 , 128×64 , and 256×128 , relaxation time $\tau = 1.0$, Reynolds number $Re = \frac{u_\infty L}{\nu} = 10$, reference length $L = N_y^f + 1$ to enable the halfway bounceback to be implemented on the half-displacement of the upper and lower surfaces, and the fixed-to-moving grid size ratio of $N_y^f / N_y^m = 2$. The analytical solution of the velocity profile is

$$u_{Ana} = u_{x,max} \left(\frac{4y}{L} - \frac{4y^2}{L^2} \right) \quad (6.32)$$

Similar to the uniform flow, the simulation starts with zero velocity and unity density in the overall domain, with the moving grid starting rotation from the first time step t_s to the final time step t_f . The final time step is implemented using stopping criteria ϵ to ensure the steady-state solution for the different grid sizes as

$$\epsilon = \frac{|u(\mathbf{x}_k, t) - u(\mathbf{x}_k, t - \Delta t)|}{u(\mathbf{x}_k, t - \Delta t)} \leq 10^{-10} \quad (6.33)$$

The numerical and analytical solutions' velocity profiles are the same, as shown in Figure 6.7. Moreover, the velocity field of the overlapped grids is shown in Figure 6.8, as there is no deviation between the fixed and moving grids on the moving-to-fixed border, ensuring the correct velocity field and the implementation of the Poiseuille flow.

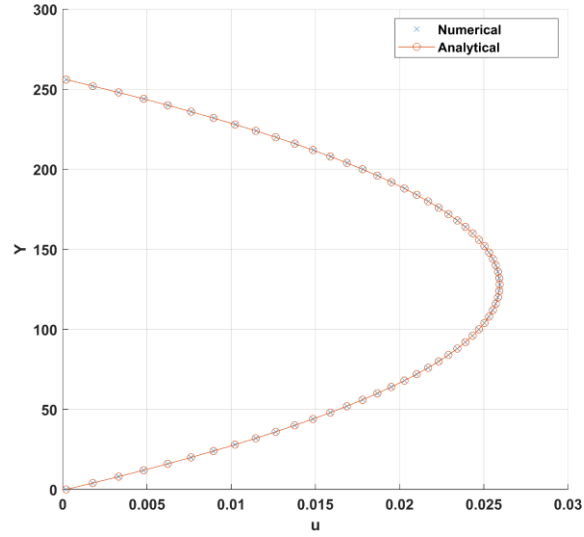


Figure 6.7 Velocity profile after the moving grid at 75% of the channel length for the test case of 128×64 at t_f .

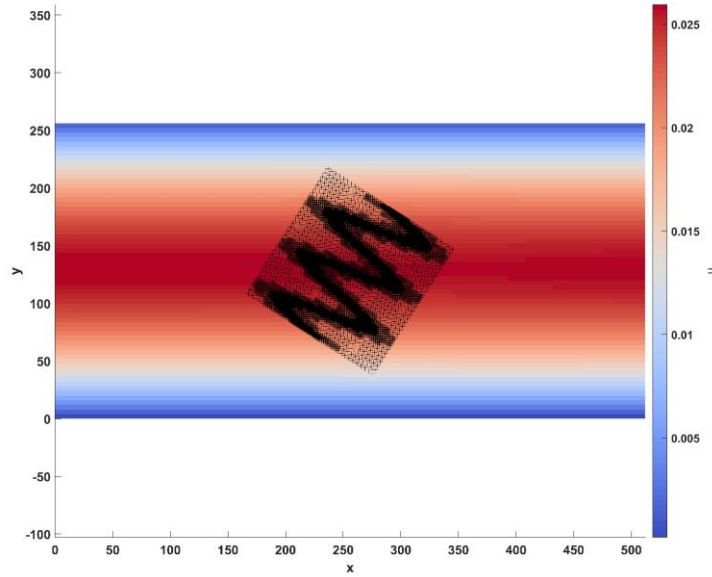


Figure 6.8 Velocity field of the overlapped grids for the test case of 128×64 at t_f .

The simulations are also tested through the error L_1^u as in Eqn. (5.41) with $\mathbf{u}_{ref}$ is the analytical solution, and the pressure-noise space-average value $\frac{\partial \bar{p}}{\partial t}$ as in Eqn. (5.44). Moreover, the L2 velocity error norm L_2^u is calculated and the corresponding order of accuracy to ensure that the method did not change the order of accuracy of the standard LBM. The L_2^u is calculated on the overall fixed grid and the empty nodes in the fixed grid are reconstructed in the postprocessing from the moving grid using bicubic interpolation.

$$L_2^u = \sqrt{\frac{\sum_{k=0}^{N-1} (\|\mathbf{u}(\mathbf{x}_k) - \mathbf{u}_{ref}(\mathbf{x}_k)\|)^2}{N_x^f N_y^f}} \quad (6.34)$$

Where N is the fixed grid nodes' number, N_x^f and N_y^f are the cell's number of the fixed grid in x and y directions, k are the fixed grid nodes, and $\mathbf{u}_{ref}$ is the velocity analytical solution.

Table 6.2. The Poiseuille flow L_1^u , $\frac{\partial \bar{p}}{\partial t}$, L_2^u , and order of accuracy at t_f .

No	$N_x^f \times N_y^f$	L_1^u	$\frac{\partial \bar{p}}{\partial t}$	L_2^u	Order of accuracy
1	32×16	2.2561e-03	6.0643e-10	1.1105e-04	
2	64×32	2.9788e-04	3.3999e-10	3.0251e-05	1.84
3	128×64	3.3645e-05	3.0966e-10	7.4543e-06	2.02
4	256×128	3.8407e-06	1.5488e-11	1.8601e-06	2.00

Table 6.2 shows that the L_1^u and $\frac{\partial \bar{p}}{\partial t}$ are decreasing with the increasing grid size, showing low error values even with low mesh size. Moreover, the order of accuracy converges to 2, which verifies the second order of accuracy in space for the present study, and the order of accuracy did not deteriorate as the bilinear interpolation is used as a predictor for a very low number of nodes.

6.5.3 Taylor-Couette flow

The Taylor-Couette flow setup examines the method when incorporating boundary rotation and a more steep velocity profile. This test case is implemented to ensure that the method works correctly with the curved boundary treatment of Guo's extrapolation method by comparing the analytical and numerical solutions velocity profiles. In this configuration, the flow occurs between two cylinders. The outer cylinder is located in the fixed grid, while the inner cylinder is positioned within the moving grid, denoted as borders 1 and 2 in Figure 6.9. a), respectively. Moreover, the inner cylinder's boundary enforces a zero velocity using the curved boundary treatment provided

by the Guo extrapolation technique [30]. As a result, the influence of the inner cylinder's rotation is only attributable to the rotation of the moving grid.

The two cylinders' centers are in the center of a 128×128 square, and the outer radius $R_{out} = 0.45H$, where H is the square edge length. Moreover, the inner circle $R_{in} = \beta R_{out}$, where β is the radii ratio, the reference length is the thickness between the two cylinders $L = R_{out} - R_{in}$, relaxation time $\tau = 1.0$, Reynolds number $Re = 100$, the angular speed $\omega = u_{max}/R_{in}$, and the fixed-to-moving grid size ratio of $N_y^f/N_y^m = 2$. The domain is initialized with zero velocity and unity density, with moving grid rotation starting in the first time step till the final time step t_f based on the stopping criteria in equation (6.33).

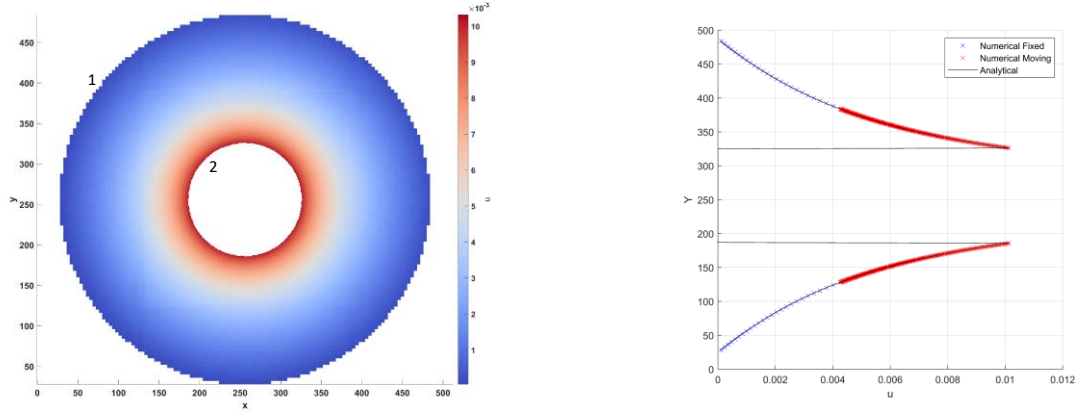
The analytical solution for the Taylor-Couette flow [89] is based on the inner and outer cylinders' radii and their corresponding velocities, as shown in equation (5.46). In this test case, the outer cylinder velocity is zero, while the inner has the maximum speed. Thus, the analytical solution could be presented as a function of the radii ratio β as shown in equation (5.47).

As described in reference [36], the analytical solution for the Taylor-Couette flow relies on the radii of the inner and outer cylinders and their respective velocities, as represented by equation (5.46). In the present study, the outer cylinder maintains a zero velocity while the inner cylinder attains the maximum speed. Consequently, the analytical solution can be expressed as a function of the radius ratio β , as in equation (5.47).

$$u_{Ana}(x, y) = u_{inner} \left(\frac{\frac{R_{out}}{r(x, y)} - \frac{r(x, y)}{R_{out}}}{\frac{R_{out}}{R_{in}} - \frac{R_{in}}{R_{out}}} \right) + u_{outer} \left(\frac{\frac{r(x, y)}{R_{in}} - \frac{R_{in}}{r(x, y)}}{\frac{R_{out}}{R_{in}} - \frac{R_{in}}{R_{out}}} \right) \quad (6.35)$$

$$u_{Ana}(x, y) = u_{max} \left(\frac{\beta}{1 - \beta^2} \right) \left(\frac{R_{out}}{r(x, y)} - \frac{r(x, y)}{R_{out}} \right) \quad (6.36)$$

The analytical solution shows that the lower the radii ratio β , the steeper the velocity profile; thus, the radii ratio $\beta = 0.3$ is selected. The numerical solution velocity field and velocity profile of the numerical and analytical solution are shown in Figure 6.9. a) and b), respectively. The figures show a good velocity profile, as the numerical and analytical velocity profiles are identical. Moreover, the L1 velocity error $e_1^u = \frac{1}{N} \sum_{k=0}^{N-1} |u(\mathbf{x}_k) - u_{Ana}(\mathbf{x}_k)|$ is calculated in both fixed and moving grids and equals $e_1^u = 4.5798E - 05$ which reveals a low value of the velocity error.



a) Overlapped grids' velocity field.

b) velocity profile of the numerical and analytical solution at the middle of the domain.

Figure 6.9 Taylor-Couette flow velocity field of the overlapped grids at t_f .

6.5.4 Flow around a rotating cylinder

The flow around a rotating cylinder aims to validate the LB-overset spatial Lagrangian-link PDF streaming method by comparing the lift and drag coefficients with the literature at different Reynolds numbers and rotational speeds. The rotating cylinder is located in the moving quadtree grid, while the fixed grid is a uniform coarse grid. Moreover, the refinement control points are designated along the perimeters and adjusted at varying layer depths, each adhering to distinct level criteria to facilitate a gradual refinement, as shown in Figure 6.10. Moreover, the moving grid streaming nodes splitting of the direct streaming, Lagrangian link, and spatial Lagrangian-link nodes is performed as shown in Figure 6.11.

The Zou and He velocity input and pressure output boundary conditions, as described in reference [85], are applied to the inlet and outlet boundary conditions. For the lower and upper surfaces, straight velocity boundary conditions are enforced using the Inamuro boundary condition method [87]. A curved boundary treatment is also employed by utilizing the extrapolation Guo technique, as outlined in reference [30]. This technique involves constructing the post-collision particle distribution function on the boundaries by extrapolating key parameters such as density, velocity, and non-equilibrium distribution function from adjacent fluid nodes.

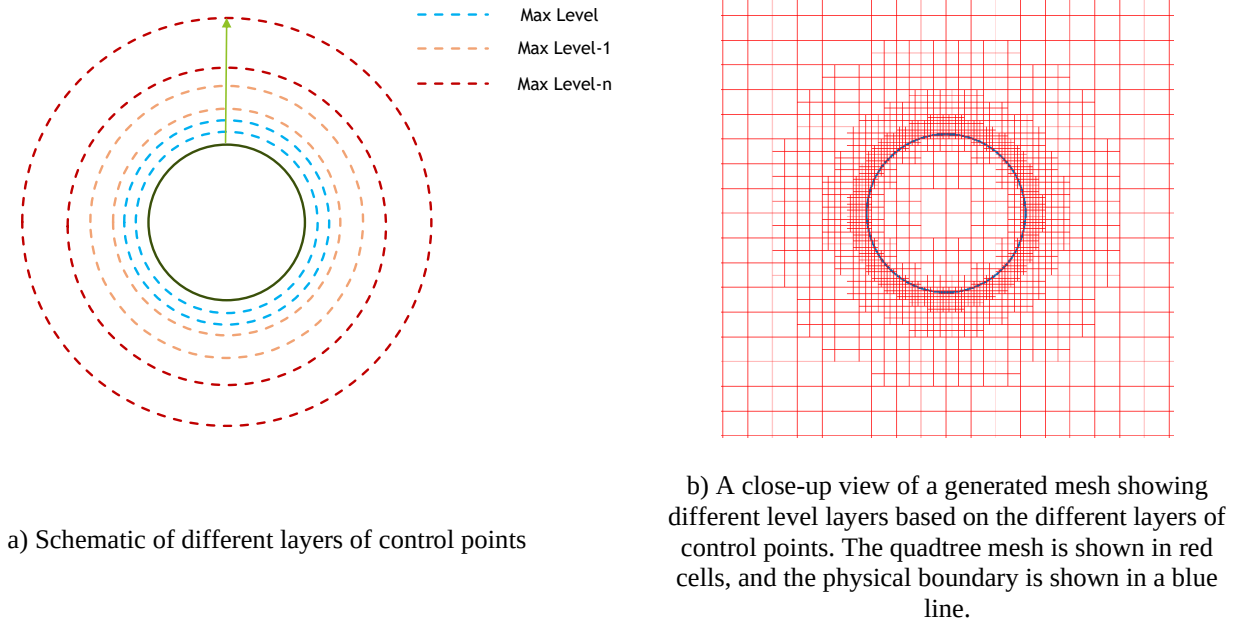


Figure 6.10 Different layers of control points for flow around a cylinder. The presented figure is coarser than the one used in the simulation for visualization purposes.

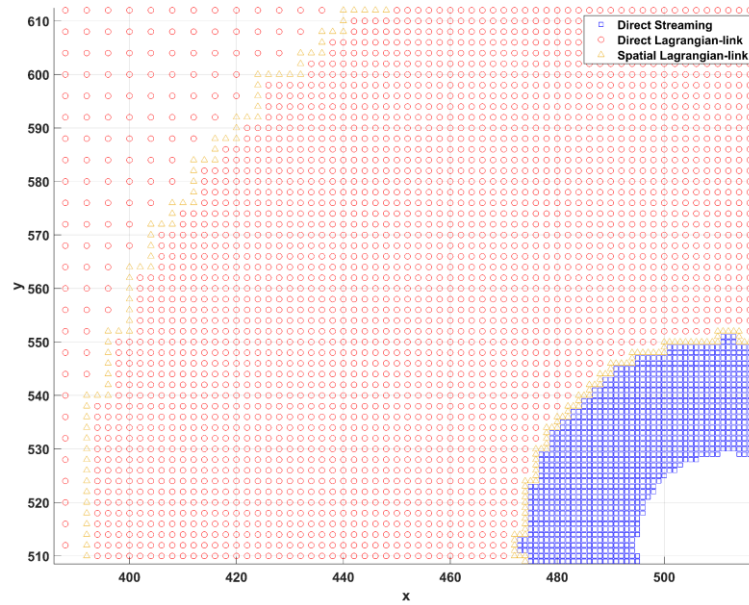


Figure 6.11 An example of the moving quadtree grid streaming nodes splitting. The blue squares are the direct streaming nodes, the red circles are the direct Lagrangian-link nodes, and the orange triangles are the spatial Lagrangian-link nodes.

The simulation starts with zero velocity and unity density in the overall domain and the total forces acting on the cylinder boundary $\mathbf{F}_B$ are calculated according to the momentum exchange theory [32] as

$$\mathbf{F}_B = \sum_{k \in B_n} \sum_{i=1}^8 \xi_i (f_i^*(\mathbf{X}_k, t) + f_i^*(\mathbf{X}_k + \xi_i \delta_t, t)) \cdot \delta_k \quad (6.37)$$

Here, B_n represents the set of boundary nodes and δ_k is equal to 1 when a node is fluid and 0 otherwise. Boundary nodes are those nodes surrounded by both fluid and solid nodes. The post-collision PDFs at these boundary nodes are denoted as $f_i^*(X_b, t)$, while $\bar{f}_i^*(X_b + \bar{\xi}_i \delta_t, t)$ are the post-collision PDFs of neighboring nodes in the opposite direction. Furthermore, the lift and drag coefficients at zero angle of attack are determined from the total force components $\mathbf{F}_B = F_x \hat{i} + F_y \hat{j}$. However, this force should be transformed to be evaluated in the inertial fixed frame $\mathbf{F}_B^f = F_x^f \hat{i} + F_y^f \hat{j}$ as :

$$\mathbf{F}_B^f = \mathcal{R}^T \mathbf{F}_B \quad (6.38)$$

$$C_L = \frac{2 F_y^f}{\rho u_\infty^2 D} \quad (6.39)$$

$$C_D = \frac{2 F_x^f}{\rho u_\infty^2 D} \quad (6.40)$$

Where D is the cylinder diameter and u_∞ is the freestream velocity. The method validation is implemented by comparing the lift and drag coefficients of the current study with previously published data. Table 6.3 shows the lift coefficient amplitude C'_L , mean drag coefficient $\bar{C}_D$ and Strouhal number $St = f L / u_\infty$ for different Reynolds number and rotational ratios, where f is the vortex shedding frequency, which agrees with the published data.

Table 6.3. The lift and drag coefficients and Strouhal number. The metric values of the Kang et al. study [94] are extracted from its figures.

Case		Re	a	$N_x^f \times N_y^f$	Ma	τ	$\bar{C}_L$	$\bar{C}_D$	C'_L	St
1	Ours, present study	100	2	512×512	0.04	0.595	5.519	0.894	0	0
	Kang et al. [94]			–	–	–	5.499	0.87	0	0
2	Ours, present study	100	1	512×512	0.04	0.595	2.507	1.282	0.372	0.167
	Kang et al. [94]			–	–	–	2.48	1.31	0.366	0.167
	Tang and Ingham [109]			–	–	–	1.973	1.008	–	–
3	Ours, present study	60	2	512×512	0.04	0.658	5.649	0.711	0	0

<i>Case</i>		<i>Re</i>	<i>a</i>	$N_x^f \times N_y^f$	<i>Ma</i>	τ	$\bar{C}_L$	$\bar{C}_D$	C_L'	<i>St</i>
	Kang et al. [94]			–	–	–	5.614	0.67	0	0
4	Ours, present study	60	1	512×512	0.08	0.657	2.56	1.226	0.131	0.139
	Kang et al. [94]			–	–	–	2.55	1.167	0.123	0.139
	Tang and Ingham [109]			–	–	–	2.249	1.219	–	–
5	Ours, present study	20	1	256×256	0.04	0.736	2.794	1.944	0	0
	Kang et al. [94]			–	–	–	2.76	–	0	0
	Badr et al. [95]			–	–	–	2.79	1.91	0	0
6	Ours, present study	20	0.5	256×256	0.04	0.736	1.379	2.08	0	0
	Ours, LB-overset without quadtree			1200×1200	0.08	0.76	1.388	2.055	0	0
	Kang et al. [94]			–	–	–	1.345	–	0	0
	Badr et al. [95]			–	–	–	1.39	2.12	0	0

The present study demonstrates strong agreement with the published data across all cases, and any disparities in these metrics were primarily attributed to the different methodologies and boundary conditions employed. For instance, Kang et al. employed an approach that addressed unsteady incompressible flow with boundary nodes conforming to the cylinder walls. In contrast, our study utilized an unsteady weakly compressible solver, and the boundary nodes were not aligned with the cylinder wall. Nevertheless, these variations did not significantly impact the results for each metric compared to the reference data.

Moreover, in examining cases 1, 2, 3, and 4, the fixed grid sizes were larger than the other cases. This increase in grid size is primarily because these cases exhibit a higher Reynolds number, necessitating an enlargement of the grid size to avoid a low relaxation time τ , near 0.5. In cases 1 and 3, there is a high rotation ratio a , which demands a reduction in the freestream velocity u_∞ to mitigate the effects of a high rotational speed $\omega = 2 u_\infty a/D$. This rotational speed reduction can be achieved by increasing the grid size, thereby allowing for a decrease in the freestream velocity while keeping the relaxation time constant and increasing the cylinder diameter D , which is related to the fixed grid height. Furthermore, cases 1, 3, 5, and 6 have flows produced a steady solution similar to the published data where the Strouhal number is zero, while cases 2 and 4 produce an

unsteady flow with vortex street with values matched with the published data. Additionally, in test case 6, a comparison was made between the results of the present study and our LB-overset simulation without quadtree (direct streaming for all nodes), which have excellent agreement across all metrics, revealing that the spatial Lagrangian-link streaming did not degrade the solution.

More investigation of the present study's effect is performed by comparing the velocity field of a bilinear PDF streaming and the spatial Lagrangian-link PDF streaming method shown in Figure 6.12 a) and b), respectively. Both simulations have $Re = 100$, rotational ratio $a = 1$, and fixed grid size 256×256 cells. So, the fixed grid height and length are scaled by the value of $2^{LVL_{max}}$. The bilinear PDF streaming is implemented by interpolating the post-collision PDF required for streaming using the bilinear interpolation only without any Lagrangian-link interpolation. In such case, the velocity profile must have a vortex street in the wake of the cylinder as the rotational ratio did not reach the steady solution rotational ratio a_s , in $Re = 100$ the $a_s \approx 2$. The bilinear PDF streaming failed to recover the accurate velocity field because it did not generate the vortex street phenomenon. This limitation can be explained by the fact that bilinear PDF streaming only captures the interpolated PDF value from a single cell, which is insufficient for accurately representing the velocity gradient. On the other hand, the spatial Lagrangian-link PDF streaming retrieved the correct velocity profile as the vortex street was developed.

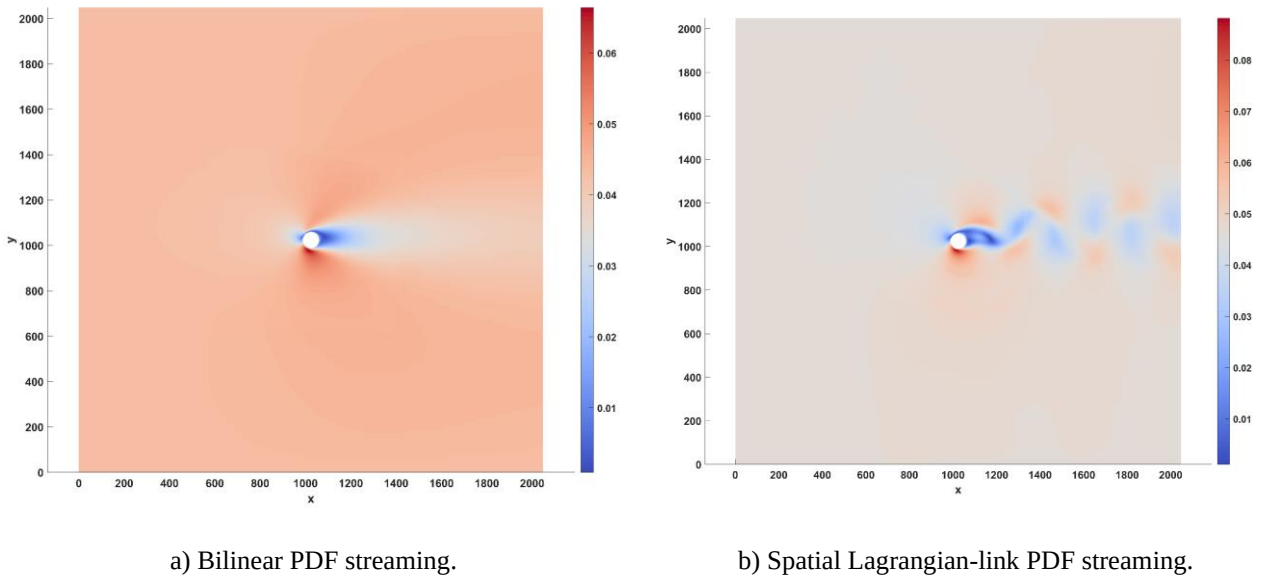


Figure 6.12 Comparison of the velocity profile using the Bilinear PDF streaming and the spatial Lagrangian-link PDF streaming for $Re = 100$ and $a = 1$. The colorbar is regarding to the velocity values.

Additionally, Figure 6.13 adds more qualitative validation by showing the effect of the rotational speed on the flow field solution steadiness. Both simulations are implemented using the LB-overset quadtree spatial Lagrangian-link method. Figure 6.13.a) shows the velocity profile for case 2, $Re = 100$ and $a = 1$, which resulted in an unsteady solution and developing a vortex street because the rotational ratio a is less than the steady solution rotational ratio $a_s \approx 2$. On the other hand, Figure 6.13.b) shows velocity profile for case 1, $Re = 100$ and $a = 2$, where a steady solution was developed because the rotational ratio reaches its steadiness value $a_s \approx 2$.

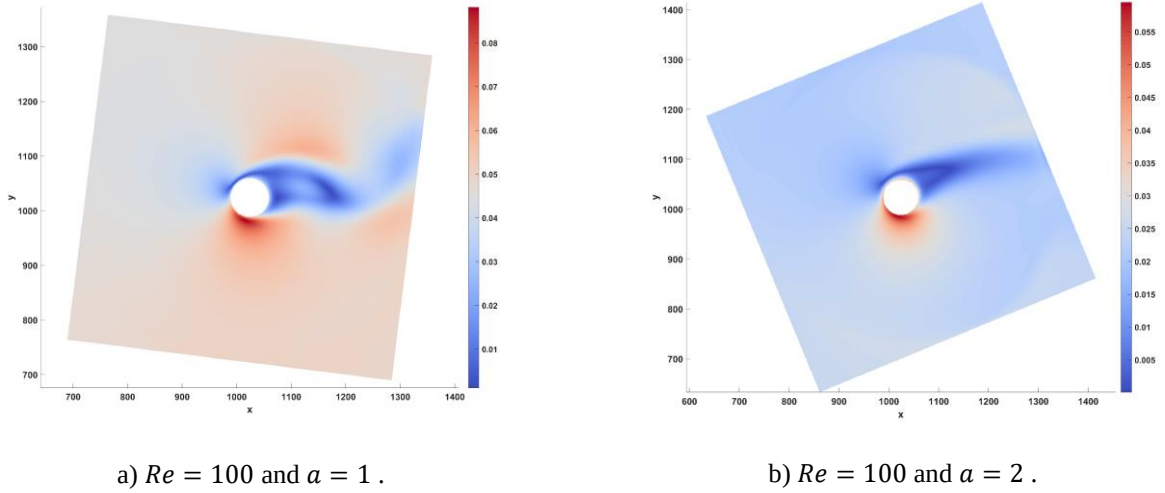
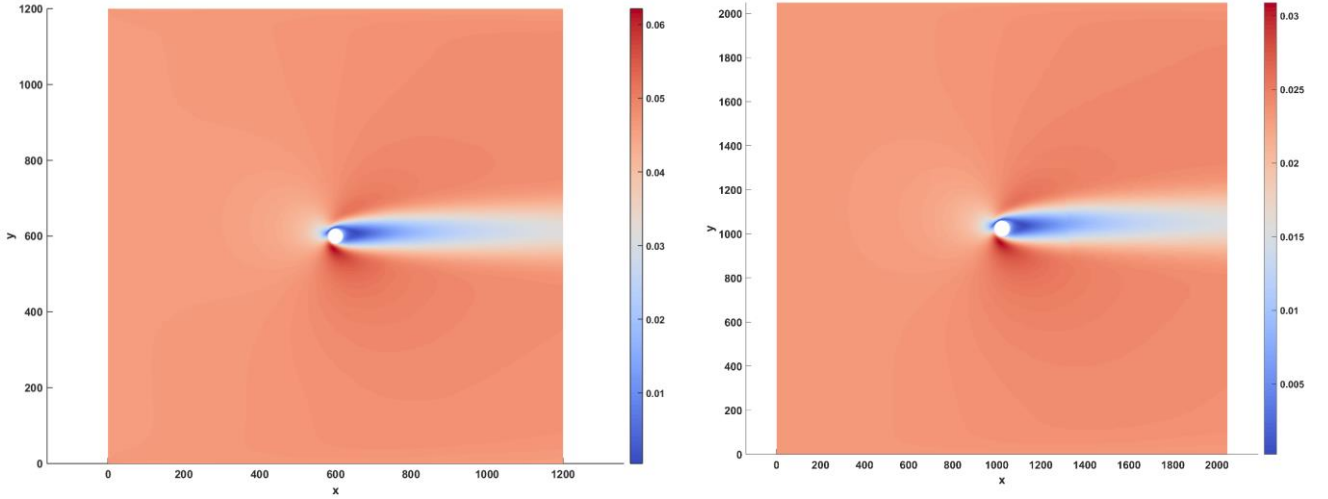


Figure 6.13 Comparison of the velocity profile of the moving grid regarding the steady and the unsteady solution depending on the rotational ratio. The color bar is regarding the velocity values.

Moreover, Figure 6.14 a) and b) show the velocity profiles of the standard LB-overset method (no quadtree mesh) and LB-overset with quadtree spatial Lagrangian-link PDF streaming technique, respectively, for $Re = 20$ and $a = 0.5$. The LB-overset method without quadtree applies direct streaming of standard LBM for all nodes in the fixed and moving grid. Both simulations retrieve a correct velocity field where the steady solution emerged. Moreover, their lift and drag coefficients, shown in Table 4.5, case 6, are quite similar, indicating that the proposed spatial Lagrangian-link technique did not negatively affect the solution compared with the LBM standard streaming. Moreover, the LB-overset method without quadtree required high computational resources as the fixed grid size $1200 \times 1200 = 1440000$ nodes and the moving grid size is $300 \times 300 = 90000$, compared with the LB-overset with a quadtree is for fixed grid $256 \times 256 = 65536$ nodes and moving grid 35500 nodes. This leads to a decrease in the total number of nodes by 1428964 nodes with a percentage of $1428964 / (65536 + 35500) =$

1414.31%. Furthermore, the LB-overset method without quadtree was not able to simulate high Reynolds number cases with high rotational speed, such as $Re = 100$ and $Re = 60$ with $a = 1.0$ on AMD Hawaii, 12288 MB GPU device due to a lack of computational resources. This is achieved by using the quadtree spatial Lagrangian-link PDF streaming technique, as it decreases the required computational resources significantly.



a) LB-overset with no quadtree.

b) LB-overset spatial Lagrangian-link PDF streaming.

Figure 6.14 Comparison of the velocity profile using the LB-overset method with and without the quadtree spatial Lagrangian-link PDF streaming for $Re = 20$ and $a = 0.5$. The colorbar is regarding the velocity values.

6.6 Conclusion

In the present study, a spatial Lagrangian-link PDF interpolation streaming technique is introduced and integrated with the LB-overset method with quadtree mesh refinement. The method is verified and validated with the previously published numerical data and does not show any degradation in the results or the order of accuracy. First, the uniform flow test case shows the correct implementation of the proposed method as no flow distortion appeared, and the pressure noise reaches the machine's accuracy, revealing that the solution was not affected by the moving grid rotation or the reconstruction of the interpolated PDFs. Then, Poiseuille flow verifies the method by retrieving a second order of accuracy in space and the correct velocity profile, showing that using the spatial Lagrangian interpolation method did not degrade the order of accuracy. Next, the Taylor-Couette flow shows the correct implementation of the present method with the curved boundary treatment of Gou's extrapolation method by retrieving the correct velocity profile. After

that, the flow around the rotating cylinder validates the proposed method by retrieving matched results with the published lift, drag, and Strouhal number data.

Moreover, using the bilinear interpolation alone for interpolating the post-collision PDFs is insufficient, as the interpolator is performed using one cell and could not capture the effect of the other surrounding cell, consequently not capturing the gradient effect. This issue does not appear in the spatial Lagrangian-link method because the spatial interpolation is performed as a predictor step on a low number of points, while the Lagrangian-link interpolation acts as a corrector step of a quadratic interpolation.

Additionally, using the quadtree mesh significantly decreases the required computational resources. Also, the spatial Lagrangian-link interpolation decreases the number of nodes required for implementing the PDF streaming than the spatial interpolations, decreasing the required computational resources and increasing the simulation efficiency. Moreover, the node-splitting technique allows the interpolations or the direct streaming to be executed depending on the nodes' requirements, which increases the simulation efficiency. Furthermore, the current approach can be employed with non-conforming quadtree rectangular grids, such as Pellerin et al.'s work [43], and can also be extended to 3D applications. This extension offers additional advantages, as it demands significantly fewer nodes for interpolation compared to the extension of spatial interpolators like bicubic methods, which decrease the required computational resources and increase the simulation efficiency.

Acknowledgment

The author, Abdallah ElSherbiny, would like to thank Professor Jean-Yves Trepanier, Department of Mechanical Engineering, Polytechnique Montréal, and Dr. Nicolas Pellerin, Aircraft Performance Engineer, National Research Council of Canada, Ottawa, for constructive criticism of the manuscript and the fruitful discussions about the present study. Moreover, the authors acknowledge the support of the Natural Sciences and Engineering Council of Canada (NSERC), [funding reference number RGPIN-2020-04512].

CHAPTER 7 GENERAL DISCUSSION

The development of the thesis involved several essential stages. Initially, it started with an introduction to the challenge of addressing unsteady flow dynamics around moving objects, offering an overview of different CFD solvers and a discussion of their respective attributes. Following this, a comprehensive literature review is articulated, exploring the LBM, mesh refinement techniques, and moving boundary treatment approaches. This examination reveals the advantages of the quadtree mesh and the LB-overset method over the mesh refinement and the moving boundary treatment approaches, respectively. Additionally, it highlights a research gap wherein previous studies did not combine the LB-overset method with the quadtree, thus indicating the potential for integrating the overset method and quadtree mesh.

Then, based on the research gap, the thesis's main objective is defined as “verifying and validating the integrated Lattice Boltzmann method with the moving overset method and the quadtree grid refinement based on the interpolated PDF streaming scheme” and the methodology to achieve this objective is discussed and classified into three main phases.

The first phase was discussed in Chapter 4, verifying and validating the standard LBM with the different boundary conditions. This phase represents the base of the second and third phases as it describes the primary data structure and the classification of the grid nodes. The grid nodes are classified into solid, fluid, and boundary nodes, which avoid storing the variables on the solid nodes, enable the execution of the curved boundary treatment on the detected boundary nodes, and implement the standard solution steps on the fluid nodes. Moreover, this phase validates and verifies the standard LBM with the curved boundary treatment using Taylor-Couette flow, flow around stationary cylinder, and flow around rotating cylinder. Furthermore, the Poiseuille flow reveals that a third order of accuracy in space was retrieved when a halfway bounceback was implemented on a straight linear boundary with the lattice system of units.

The second phase was illustrated in Chapter 5, which is an intermediate phase as the overset method is integrated with the LBM before adding the quadtree mesh. This phase discussed the LB-overset method and implemented a comparison between different interpolators required for reconstructing the macroscopic variables on the grids' borders. These interpolators cover a wide range of interpolation methods based on regular and irregular grids, interpolation nodes' numbers,

and different orders of accuracy. These interpolators are tested through the uniform flow, Poiseuille flow, Taylor-Couette flow, high Reynolds advection vortex in uniform flow, flow around stationary cylinder, and flow around rotating cylinder. The uniform flow is selected to detect any deviation in the flow due to the moving grid rotation, while the Poiseuille and Taylor-Couette flow add more complex velocity gradients to the simulations. These three test cases dropped some proposed interpolators; however, they could not distinguish between the bilinear and bicubic interpolators as the flow is still steady, and the interpolations are applied to the macroscopic variables. Thus, the high Reynolds advection vortex in uniform flow is implemented, showing that the bicubic interpolator reserves the accurate velocity profile. Then, the flow around stationary and rotating cylinder test cases is implemented to validate the LB-overset method.

The third and final phase combines the LB-overset method with a non-conforming quadtree mesh using a spatial Lagrangian-link interpolated PDF streaming scheme, discussed in Chapter 7. The LB-overset method with uniform grids relies on two grids with the same spacing, demanding extremely high computational resources. To mitigate this resource-intensive aspect, a quadtree mesh is employed to reduce the number of nodes. However, using the quadtree introduces difficulties related to varying cell levels and spatial displacements, which are solved using the interpolated PDF streaming technique. However, the literature introduced the reconstruction of the PDFs using spatial interpolations, which adds more complexity and requires high computational resources. This issue is solved by exploiting the LBM Lagrangian nature as the spatial Lagrangian-link interpolation scheme is introduced.

This interpolation method started with spatial interpolation as a predictor step. Subsequently, this spatial predictor-interpolated PDF was utilized for a Lagrangian-link corrector interpolation. The node-splitting technique was also introduced, which improved the proposed interpolation scheme's efficiency as most of the domain's nodes required direct streaming or direct 1D Lagrangian interpolation. Moreover, the second phase of comparing various interpolation methods for LB-overset grid communication did not include the implementation of spatial Lagrangian-link interpolation. This omission is because spatial Lagrangian-link interpolation is associated with the interpolation of microscopic variables mentioned in the PDF, while LB-overset grids communication is implemented using the macroscopic variables. Notably, the method maintains its order of accuracy as a second order, and the results from simulating flow around a rotating cylinder validated the approach as they matched with previously published data.

The suggested approach maintains the LBM parallelization advantage by relying on the local collision of PDFs on active nodes. This characteristic can be underscored by applying the developed technique for splitting streaming nodes. Additionally, extending the method to accommodate 3D flow is straightforward, as it builds upon the 1D Lagrangian interpolation. This results in a reduction in computational time and complexity, as only the determination of one unknown PDF on the link is needed using three nodes rather than 16 nodes in the case of bicubic interpolation. Additionally, a bilinear interpolation can still be employed for the spatial predictor step, utilizing the four nodes located on the plane where the PDF appears.

The computational cost is determined by evaluating the required average computational time per time step t_c for the LB-overset method with various interpolators and the quadtree LB-overset method with spatial-Lagrangian link PDF interpolated scheme (with and without streaming nodes splitting technique), as shown in Table 7.1. All results are obtained for the same test case of solving the flow around a rotating cylinder with $Re = 20$ and $a = 0.5$. The average computational time is calculated by starting the clock at the first step in the time loop and ending the clock at the end of the time loop just before any post-processing calculations. Then, the simulation continues to run until the average computational time value stabilizes. Moreover, the million lattice updates per second MLUPS is calculated to indicate the required processing time per node. MLUPS is calculated by dividing the total number of nodes N_{tot} by the average computational time per time step t_c .

Table 7.1. The average computational time per time step t_c , the total number of nodes N_{tot} , and MLUPS for the LB-overset and quadtree LB-overset methods.

Method	LB-overset					Quadtree LB-overset without nodes splitting	Quadtree LB-overset with nodes splitting
	IDW4	Bilinear	IDW16	Bicubic convolution	Bicubic		
t_c [Sec]	1.298	1.309	1.318	1.329	1.455	1.083	0.876
N_{tot} [Million]	1.440	1.440	1.440	1.440	1.440	0.101	0.101
MLUPS	1.1094	1.1001	1.0926	1.0835	0.9897	0.0933	0.1251

The LB-overset method, utilizing bicubic interpolation, has the highest computational cost at $t_c = 1.455$ due to the necessity of solving a 16-equation matrix. In contrast, the other interpolators exhibit lower computational costs, with IDW4 has the lowest value. Conversely, the quadtree LB-overset approach, employing a spatial-Lagrangian link PDF interpolated scheme with the

streaming nodes splitting technique, significantly lowers computational cost with $t_c = 0.876$ by reducing the number of nodes and relying on 1D Lagrangian interpolation. For instance, if we have a simulation of 200000 iterations, the time saved using the quadtree LB-overset is $0.4 \times 200000 = 80000 \text{ Sec} \approx 22.22 \text{ hour}$. This reduction mainly appears because of the significant decrease in the total number of nodes. Furthermore, the total number of nodes for the quadtree LB-overset is considerably reduced compared with the LB-overset, significantly reducing the required memory and computational resources. This demonstrates that the proposed method not only reduces the computational resources in terms of memory but also enhances the simulation efficiency by decreasing the computational cost. However, the MLPUS for the quadtree LB-overset is significantly low because of the additional process of interpolated PDF streaming, where most of the streaming PDFs on most nodes require interpolations. In contrast, the MLUPS for the LB-overset method with no quadtree is much higher because all nodes perform direct streaming without interpolation.

Additionally, the streaming nodes' splitting technique improves the average computational time compared with the quadtree LB-overset without streaming-splitting technique with average computational time saved per time step $\Delta t_c = 0.2$, which could save about 11 *hour* for a simulation of 200000 iterations. Furthermore, the MLUPS of the streaming nodes splitting technique is higher than without using it, further improving the simulation efficiency.

CHAPTER 8 CONCLUSION AND RECOMMENDATIONS

8.1 Conclusion

The primary goal of the thesis, which was to verify and code-to-code verify the integration of the Lattice Boltzmann method with the moving overset method and the quadtree grid refinement using the interpolated PDF streaming scheme, has been successfully accomplished. This verification and code-to-code verification process involved three key steps. Firstly, the verification and code-to-code verification of the standard LBM with both linear and curved boundaries were conducted. Subsequently, the investigation into integrating the overset method with LBM was carried out, exploring nine different communication interpolators. Finally, the verification and code-to-code verification of the application of the quadtree spatial Lagrangian-link interpolated PDF streaming method with LB-overset demonstrated a second-order spatial accuracy.

The LB-overset method yielded exceptional results in solving problems involving moving boundaries, producing accurate velocity profiles without any distortions, and exhibiting minimal pressure noise resulting from the rotation of the moving grid. However, its practicality was constrained by the significant computational resources required, owing to using two grids. This challenge was effectively addressed by transforming the moving grid into a quadtree grid while maintaining the fixed grid as a uniform grid with the highest spatial displacement. This transition to a quadtree structure substantially reduced the computational resources needed, with some cases experiencing a remarkable reduction of up to 1414.31%.

Furthermore, the spatial Lagrangian-link interpolation technique reduced the number of nodes necessary for implementing PDF streaming compared to published spatial interpolations. This node reduction leads to decreased computational resource requirements and enhanced simulation efficiency. Additionally, the node-splitting technique allowed for the execution of interpolations or direct streaming depending on the specific needs of the nodes, further improving simulation efficiency.

It's worth noting that this study's approach can be extended to non-conforming quadtree rectangular grids, similar to the work done by Pellerin et al. [43], and can be further expanded to three-dimensional applications due to the inherent Lagrangian nature of the LBM. This extension

offers several advantages, such as a significant reduction in the number of nodes required for interpolation compared to the extension of spatial interpolators like the bicubic method. This, in turn, results in reduced computational resource requirements and increased simulation efficiency.

To sum it up, the main contribution of the thesis is introducing an integrated method of the quadtree LB-overset based on the interpolated PDF streaming technique, which fills the research gap presented in Chapter 3. In contrast, the secondary contributions are introducing the spatial Lagrangian-link approach, the nodes splitting (decomposition) technique, and investigating the interpolators' effect on the LB-overset.

8.2 Limitations

Regarding the physical limitations, the method can be applied to rigid body FSI, not to deformable FSI. The limitation to the rigid body FSI arises from the fixation of the moving boundary to the moving grid in the LB-overset method, which prevents alterations or deformations of the boundary.

Regarding the numerical limitations, this study solved unsteady incompressible 2D flow as it aimed to verify and validate the method. Thus, it was limited to using low Mach and Reynolds numbers in 2D flows. However, it can be extended to 3D and modified to add the compressibility effect, such as using the thermal hybrid LBM [110].

8.3 Future research and recommendations

The study could be extended in two main areas: the method's code improvement and the method's physics improvement.

Method's physics improvement:

- I. Extension to the non-conforming rectangular quadtree with the LB-overset method, as the spatial Lagrangian-link interpolation based can apply the predictor step on a rectangular grid using a bilinear interpolation.
- II. Extension to 3D flows is applicable due to the LBM Lagrangian nature, and it would require low computational resources compared with other methods.
 - I. Add the compressibility effect to the LBM solver to add the availability of simulating high Mach numbers.
 - II. As an LBM solver, the availability to be extended to a multiphase or a multicomponent flow exists.

- III. Apply the method to real applications such as wind turbines, wing control surfaces, or movable wing vertical takeoff and landing, as the thesis only aimed to verify and validate the method.

Method's code improvement:

- I. Further improving MLUPS for quadtree LB-overset method.
- II. Modify the nodes' splitting algorithm based on the direction instead of nodes surrounding cell levels.
- III. The study implementation was designed for parallel computing in one GPU device; in future work, parallel computing on multiple GPU devices will be considered.

REFERENCES

- [1] R. Johnson, M. Stanek, and J. Grove, "Store separation trajectory deviations due to unsteady weapons bay aerodynamics," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, 2008, p. 188.
- [2] T. Wang and F. Coton, "Prediction of the unsteady aerodynamic characteristics of horizontal axis wind turbines including three-dimensional effects," *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, vol. 214, no. 5, pp. 385-400, 2000.
- [3] B. Xu, T. Wang, Y. Yuan, and J. Cao, "Unsteady aerodynamic analysis for offshore floating wind turbines under different wind conditions," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 373, no. 2035, p. 20140080, 2015.
- [4] D. Marten, J. Wendler, G. Pechlivanoglou, C. N. Nayeri, and C. O. Paschereit, "QBLADE: an open source tool for design and simulation of horizontal and vertical axis wind turbines," *Int. J. Emerging Technol. Adv. Eng*, vol. 3, no. 3, pp. 264-269, 2013.
- [5] A. Krzysiak and J. Narkiewicz, "Aerodynamic loads on airfoil with trailing-edge flap pitching with different frequencies," *Journal of aircraft*, vol. 43, no. 2, pp. 407-418, 2006.
- [6] Z. Yan, H. E. Taha, and M. R. Hajj, "Geometrically-exact unsteady model for airfoils undergoing large amplitude maneuvers," *Aerospace Science and Technology*, vol. 39, pp. 293-306, 2014.
- [7] B. G. Van der Wall and J. G. Leishman, "The influence of variable flow velocity on unsteady airfoil behavior," 1992.
- [8] O. K. G. Tietjens and L. Prandtl, *Applied hydro-and aeromechanics: based on lectures of L. Prandtl*. Courier Corporation, 1957.
- [9] I. E. Garrick, "On some reciprocal relations in the theory of nonstationary flows," 1938.

- [10] M. Zakaria, H. Taha, and M. Hajj, "Measurement and modeling of lift enhancement on plunging airfoils: A frequency response approach," *Journal of Fluids and Structures*, vol. 69, pp. 187-208, 2017.
- [11] Z. Rusong, W. Fei, L. Ping, and H. Xuhui, "The continuous dynamic simulation test of the Captive Trajectory System used in 2.4 m Transonic Wind Tunnel," in *2011 Chinese Control and Decision Conference (CCDC)*, 2011: IEEE, pp. 1240-1244.
- [12] Z. G. Toor, T. I. Khan, O. Khan, B. Mufti, and J. Masud, "Comparative Analysis of Store Release Trajectory using Numerical and Experimental Techniques," in *AIAA Scitech 2019 Forum*, 2019, p. 0846.
- [13] M. Shirzadi, P. A. Mirzaei, and Y. Tominaga, "CFD analysis of cross-ventilation flow in a group of generic buildings: Comparison between steady RANS, LES and wind tunnel experiments," in *Building Simulation*, 2020, vol. 13, no. 6: Springer, pp. 1353-1372.
- [14] D. A. Perumal and A. K. Dass, "Simulation of Incompressible Flows in Two-Sided Lid-Driven Square Cavities.: Part I-FDM," *CFD Letters*, vol. 2, no. 1, pp. 13-24, 2010.
- [15] D. A. Perumal, "Simulation of flow in Two-Sided Lid-Driven deep cavities by finite difference method," *Journal of Applied Science in the Thermodynamics and Fluid Mechanics*, vol. 6, no. 1, pp. 1-6, 2012.
- [16] D. A. Perumal and A. K. Dass, "A Review on the development of lattice Boltzmann computation of macro fluid flows and heat transfer," *Alexandria Engineering Journal*, vol. 54, no. 4, pp. 955-971, 2015.
- [17] C. G. Soares, *Advances in Renewable Energies Offshore: Proceedings of the 3rd International Conference on Renewable Energies Offshore (RENEW 2018), October 8-10, 2018, Lisbon, Portugal*. CRC Press, 2018.
- [18] <https://www.spheric-sph.org/grand-challenges> (accessed).
- [19] X. He and L.-S. Luo, "A priori derivation of the lattice Boltzmann equation," *Physical Review E*, vol. 55, no. 6, p. R6333, 1997.
- [20] P. J. Morrison, "Hamiltonian description of the ideal fluid," *Reviews of modern physics*, vol. 70, no. 2, p. 467, 1998.
- [21] S. Dong, K. Otsuka, and K. Makihara, "Hamiltonian formulation with reduced variables for flexible multibody systems under linear constraints: Theory and experiment," *Journal of Sound and Vibration*, vol. 547, p. 117535, 2023.
- [22] C. Vollaire, L. Nicolas, and A. Nicolas, "Parallel computing for the finite element method," *The European Physical Journal-Applied Physics*, vol. 1, no. 3, pp. 305-314, 1998.
- [23] J. Zhou and L. S. Turng, "Three-dimensional numerical simulation of injection mold filling with a finite-volume method and parallel computing," *Advances in Polymer Technology: Journal of the Polymer Processing Institute*, vol. 25, no. 4, pp. 247-258, 2006.
- [24] B. Diskin and J. Thomas, "Effects of mesh regularity on accuracy of finite-volume schemes," in *50th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*, 2012, p. 609.

- [25] T. Inamuro, "Lattice Boltzmann methods for moving boundary flows," *Fluid Dynamics Research*, vol. 44, no. 2, p. 024001, 2012.
- [26] S. Chen and G. D. Doolen, "Lattice Boltzmann method for fluid flows," *Annual review of fluid mechanics*, vol. 30, no. 1, pp. 329-364, 1998.
- [27] M. Hofmann, D. A. Wolf-Gladrow, T. Takahashi, S. C. Sutherland, K. D. Six, and E. Maier-Reimer, "Stable carbon isotope distribution of particulate organic matter in the ocean: a model study," *Marine Chemistry*, vol. 72, no. 2-4, pp. 131-150, 2000.
- [28] T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, and E. M. Viggien, "The lattice Boltzmann method," *Springer International Publishing*, vol. 10, no. 978-3, pp. 4-15, 2017.
- [29] P. Nagar, F. Song, L. Zhu, and L. Lin, "LBM-IB: A parallel library to solve 3D fluid-structure interaction problems on manycore systems," in *2015 44th International Conference on Parallel Processing*, 2015: IEEE, pp. 51-60.
- [30] Z. Guo, C. Zheng, and B. Shi, "An extrapolation method for boundary conditions in lattice Boltzmann method," *Physics of fluids*, vol. 14, no. 6, pp. 2007-2010, 2002.
- [31] R. Mei, L.-S. Luo, and W. Shyy, "An accurate curved boundary treatment in the lattice Boltzmann method," *Journal of computational physics*, vol. 155, no. 2, pp. 307-330, 1999.
- [32] R. Mei, D. Yu, W. Shyy, and L.-S. Luo, "Force evaluation in the lattice Boltzmann method involving curved geometry," *Physical Review E*, vol. 65, no. 4, p. 041203, 2002.
- [33] K. Hejranfar, M. H. Saadat, and S. Taheri, "High-order weighted essentially nonoscillatory finite-difference formulation of the lattice Boltzmann method in generalized curvilinear coordinates," *Physical Review E*, vol. 95, no. 2, p. 023314, 2017.
- [34] A. Ghaffarian and K. Hejranfar, "An implicit dual-time stepping spectral difference lattice Boltzmann method for simulation of viscous compressible flows on structured meshes," *Meccanica*, vol. 54, no. 10, pp. 1561-1581, 2019.
- [35] A. Karbalaie and K. Hejranfar, "A Central Difference Finite Volume Lattice Boltzmann Method for Simulation of 2D Inviscid Compressible Flows on Triangular Meshes," in *ASME International Mechanical Engineering Congress and Exposition*, 2018, vol. 52101: American Society of Mechanical Engineers, p. V007T09A025.
- [36] K. Hejranfar and A. Ghaffarian, "A high-order accurate unstructured spectral difference lattice Boltzmann method for computing inviscid and viscous compressible flows," *Aerospace Science and Technology*, vol. 98, p. 105661, 2020.
- [37] K. Hejranfar, H. H. Nasab, and M. H. Azampour, "Arbitrary Lagrangian-Eulerian unstructured finite-volume lattice-Boltzmann method for computing two-dimensional compressible inviscid flows over moving bodies," *Physical Review E*, vol. 101, no. 2, p. 023308, 2020.
- [38] N. Pellerin, S. Leclaire, and M. Reggio, "Solving incompressible fluid flows on unstructured meshes with the lattice Boltzmann flux solver," *Engineering Applications of Computational Fluid Mechanics*, vol. 11, no. 1, pp. 310-327, 2017.
- [39] O. Filippova and D. Hänel, "Boundary-fitting and local grid refinement for lattice-BGK models," *International Journal of Modern Physics C*, vol. 9, no. 08, pp. 1271-1279, 1998.

- [40] Y. Peng, C. Shu, Y.-T. Chew, X. Niu, and X.-Y. Lu, "Application of multi-block approach in the immersed boundary–lattice Boltzmann method for viscous fluid flows," *Journal of computational physics*, vol. 218, no. 2, pp. 460-478, 2006.
- [41] S. Mahmoudi, S. Ayatollahi, S. Jamshidi, and A. Raoof, "Investigation on reactive flow through porous media by quadtree Lattice Boltzmann," *Journal of Natural Gas Science and Engineering*, vol. 104, p. 104674, 2022.
- [42] B. Crouse, E. Rank, M. Krafczyk, and J. Tölke, "A LB-based approach for adaptive flow simulations," *International Journal of Modern Physics B*, vol. 17, no. 01n02, pp. 109-112, 2003.
- [43] N. Pellerin, S. Leclaire, and M. Reggio, "An interpolation-based lattice Boltzmann method for non-conforming orthogonal meshes," *Computers & Mathematics with Applications*, vol. 100, pp. 152-166, 2021.
- [44] P. Valero-Lara and J. Jansson, "A non-uniform Staggered Cartesian grid approach for Lattice-Boltzmann method," *Procedia Computer Science*, vol. 51, pp. 296-305, 2015.
- [45] Z.-G. Feng and E. E. Michaelides, "The immersed boundary-lattice Boltzmann method for solving fluid–particles interaction problems," *Journal of computational physics*, vol. 195, no. 2, pp. 602-628, 2004.
- [46] S. Tao, Q. He, B. Chen, and S. Huang, "A distribution function correction-based immersed boundary-lattice Boltzmann method with truly second-order accuracy for fluid-solid flows," *arXiv preprint arXiv:1803.09380*, 2018.
- [47] H. Xiaochuan, C. JianQiang, W. Shoushui, and Y. Ting, "Lattice Boltzmann method with immersed boundary method for simulating RBC movement in micro-vessel," in *2010 8th World Congress on Intelligent Control and Automation*, 2010: IEEE, pp. 1953-1958.
- [48] L. Zhu *et al.*, "An immersed boundary method based on the lattice Boltzmann approach in three dimensions, with application," *Computers & Mathematics with Applications*, vol. 61, no. 12, pp. 3506-3518, 2011.
- [49] Z. Liu, H. Liu, D. Huang, and L. Zhou, "The Immersed Boundary-Lattice Boltzmann Method Parallel Model for Fluid-Structure Interaction on Heterogeneous Platforms," *Mathematical Problems in Engineering*, vol. 2020, 2020.
- [50] D. Wang and P. Cheng, "Nanoparticles deposition patterns in evaporating nanofluid droplets on smooth heated hydrophilic substrates: A 2D immersed boundary-lattice Boltzmann simulation," *International Journal of Heat and Mass Transfer*, vol. 168, p. 120868, 2021.
- [51] Y. Wang, C. Shu, L. Yang, and Y. Sun, "On the immersed boundary-lattice Boltzmann simulations of incompressible flows with freely moving objects," *International Journal for Numerical Methods in Fluids*, vol. 83, no. 4, pp. 331-350, 2017.
- [52] H. Yu *et al.*, "Mass-conserved volumetric lattice Boltzmann method for complex flows with willfully moving boundaries," *Physical Review E*, vol. 89, no. 6, p. 063304, 2014.
- [53] M. Meldi, E. Vergnault, and P. Sagaut, "An arbitrary Lagrangian–Eulerian approach for the simulation of immersed moving solids with Lattice Boltzmann Method," *Journal of Computational Physics*, vol. 235, pp. 182-198, 2013.

- [54] R. Zhang *et al.*, "Lattice Boltzmann approach for local reference frames," *Communications in Computational Physics*, vol. 9, no. 5, pp. 1193-1205, 2011.
- [55] P. Lallemand and L.-S. Luo, "Lattice Boltzmann equation with Overset method for moving objects in two-dimensional flows," *Journal of Computational Physics*, vol. 407, p. 109223, 2020.
- [56] M. Bahlali, H. Yoo, J. Favier, and P. Sagaut, "A lattice Boltzmann direct coupling overset approach for the moving boundary problem," *Physics of Fluids*, vol. 33, no. 5, p. 053607, 2021.
- [57] X. He and L.-S. Luo, "Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation," *Physical review E*, vol. 56, no. 6, p. 6811, 1997.
- [58] P. J. Dellar, "Incompressible limits of lattice Boltzmann equations using multiple relaxation times," *Journal of Computational Physics*, vol. 190, no. 2, pp. 351-370, 2003.
- [59] M. Geier, A. Greiner, and J. G. Korvink, "Cascaded digital lattice Boltzmann automata for high Reynolds number flow," *Physical Review E*, vol. 73, no. 6, p. 066705, 2006.
- [60] J. Latt and B. Chopard, "Lattice Boltzmann method with regularized pre-collision distribution functions," *Mathematics and Computers in Simulation*, vol. 72, no. 2-6, pp. 165-168, 2006.
- [61] H. Yoo, M. Bahlali, J. Favier, and P. Sagaut, "A hybrid recursive regularized lattice Boltzmann model with overset grids for rotating geometries," *Physics of Fluids*, vol. 33, no. 5, p. 057113, 2021.
- [62] O. Malaspinas, "Increasing stability and accuracy of the lattice Boltzmann scheme: recursivity and regularization," *arXiv preprint arXiv:1505.06900*, 2015.
- [63] C. Coreixas, G. Wissocq, G. Puigt, J.-F. Boussuge, and P. Sagaut, "Recursive regularization step for high-order lattice Boltzmann methods," *Physical Review E*, vol. 96, no. 3, p. 033306, 2017.
- [64] X. He and G. D. Doolen, "Lattice Boltzmann method on a curvilinear coordinate system: Vortex shedding behind a circular cylinder," *Physical Review E*, vol. 56, no. 1, p. 434, 1997.
- [65] X. He and G. Doolen, "Lattice Boltzmann method on curvilinear coordinates system: flow around a circular cylinder," *Journal of Computational Physics*, vol. 134, no. 2, pp. 306-315, 1997.
- [66] R. Mei and W. Shyy, "On the finite difference-based lattice Boltzmann method in curvilinear coordinates," *Journal of Computational Physics*, vol. 143, no. 2, pp. 426-448, 1998.
- [67] K. Hejranfar and E. Ezzatneshan, "Implementation of a high-order compact finite-difference lattice Boltzmann method in generalized curvilinear coordinates," *Journal of Computational Physics*, vol. 267, pp. 28-49, 2014.
- [68] Y. Li, E. J. LeBoeuf, and P. Basu, "Least-squares finite-element scheme for the lattice Boltzmann method on an unstructured mesh," *Physical Review E*, vol. 72, no. 4, p. 046711, 2005.

- [69] C. S. Peskin, "Numerical analysis of blood flow in the heart," *Journal of computational physics*, vol. 25, no. 3, pp. 220-252, 1977.
- [70] J. Wu and C. Shu, "An improved immersed boundary-lattice Boltzmann method for simulating three-dimensional incompressible flows," *Journal of Computational Physics*, vol. 229, no. 13, pp. 5022-5042, 2010.
- [71] K. Suzuki and T. Inamuro, "A higher-order immersed boundary-lattice Boltzmann method using a smooth velocity field near boundaries," *Computers & fluids*, vol. 76, pp. 105-115, 2013.
- [72] J. Qin, E. M. Kolahdouz, and B. E. Griffith, "An immersed interface-lattice Boltzmann method for fluid-structure interaction," *Journal of Computational Physics*, vol. 428, p. 109807, 2021.
- [73] C. Wang and H. Tang, "Influence of complex driving motion on propulsion performance of a heaving flexible foil," *Bioinspiration & biomimetics*, vol. 14, no. 1, p. 016011, 2018.
- [74] H.-Z. Yuan, X.-D. Niu, S. Shu, M. Li, and H. Yamaguchi, "A momentum exchange-based immersed boundary-lattice Boltzmann method for simulating a flexible filament in an incompressible flow," *Computers & Mathematics with Applications*, vol. 67, no. 5, pp. 1039-1056, 2014.
- [75] M. Jarkowski, M. Woodgate, G. Barakos, and J. Rokicki, "Towards consistent hybrid overset mesh methods for rotorcraft CFD," *International Journal for Numerical Methods in Fluids*, vol. 74, no. 8, pp. 543-576, 2014.
- [76] Y. Li, K.-J. Paik, T. Xing, and P. M. Carrica, "Dynamic overset CFD simulations of wind turbine aerodynamics," *Renewable Energy*, vol. 37, no. 1, pp. 285-298, 2012.
- [77] A. Wissink, J. Sitaraman, V. Sankaran, D. Mavriplis, and T. Pulliam, "A multi-code python-based infrastructure for overset CFD with adaptive cartesian grids," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, 2008, p. 927.
- [78] E. Atta, "Component-adaptive grid interfacing," in *19th Aerospace Sciences Meeting*, 1981, p. 382.
- [79] H. Hadzic, *Development and application of finite volume method for the computation of flows around moving bodies on unstructured, overlapping grids*. Technische Universität Hamburg, 2006.
- [80] A. Skillen, "The overset grid method, applied to the solution of the incompressible Navier-Stokes equations in two and three spatial dimensions," University of Manchester, 2012.
- [81] H. Asmuth, "Development of overset strategies for LBM-based flow solvers," Technische Universität Hamburg, 2019.
- [82] Y. Li, "An improved volumetric LBM boundary approach and its extension for sliding mesh simulation," 2011.
- [83] E. K. Far, M. Geier, and M. Krafczyk, "Simulation of rotating objects in fluids with the cumulant lattice Boltzmann model on sliding meshes," *Computers & Mathematics with Applications*, vol. 79, no. 1, pp. 3-16, 2020.

- [84] J. Tölke and M. Krafczyk, "Second order interpolation of the flow field in the lattice Boltzmann method," *Computers & Mathematics with Applications*, vol. 58, no. 5, pp. 898-902, 2009.
- [85] Q. Zou and X. He, "On pressure and velocity boundary conditions for the lattice Boltzmann BGK model," *Physics of fluids*, vol. 9, no. 6, pp. 1591-1598, 1997.
- [86] J. Latt, B. Chopard, O. Malaspinas, M. Deville, and A. Michler, "Straight velocity boundaries in the lattice Boltzmann method," *Physical Review E*, vol. 77, no. 5, p. 056703, 2008.
- [87] T. Inamuro, M. Yoshino, and F. Ogino, "A non-slip boundary condition for lattice Boltzmann simulations," *Physics of fluids*, vol. 7, no. 12, pp. 2928-2930, 1995.
- [88] D. Alciatore and R. Miranda, "A winding number and point-in-polygon algorithm," *Glaxo Virtual Anatomy Project Research Report, Department of Mechanical Engineering, Colorado State University*, 1995.
- [89] G. I. Taylor, "VIII. Stability of a viscous liquid contained between two rotating cylinders," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 223, no. 605-615, pp. 289-343, 1923.
- [90] A. L. E. Silva, A. Silveira-Neto, and J. Damasceno, "Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method," *Journal of Computational Physics*, vol. 189, no. 2, pp. 351-370, 2003.
- [91] Y.-H. Tseng and J. H. Ferziger, "A ghost-cell immersed boundary method for flow in complex geometry," *Journal of computational physics*, vol. 192, no. 2, pp. 593-623, 2003.
- [92] M.-C. Lai and C. S. Peskin, "An immersed boundary method with formal second-order accuracy and reduced numerical viscosity," *Journal of computational Physics*, vol. 160, no. 2, pp. 705-719, 2000.
- [93] P. A. Berthelsen and O. M. Faltinsen, "A local directional ghost cell approach for incompressible viscous flow problems with irregular boundaries," *Journal of computational physics*, vol. 227, no. 9, pp. 4354-4397, 2008.
- [94] S. Kang, H. Choi, and S. Lee, "Laminar flow past a rotating circular cylinder," *Physics of Fluids*, vol. 11, no. 11, pp. 3312-3321, 1999.
- [95] H. Badr, S. Dennis, and P. Young, "Steady and unsteady flow past a rotating circular cylinder at low Reynolds numbers," *Computers & Fluids*, vol. 17, no. 4, pp. 579-609, 1989.
- [96] P. L. Bhatnagar, E. P. Gross, and M. Krook, "A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems," *Physical review*, vol. 94, no. 3, p. 511, 1954.
- [97] S. Chapman and T. G. Cowling, *The mathematical theory of non-uniform gases: an account of the kinetic theory of viscosity, thermal conduction and diffusion in gases*. Cambridge university press, 1990.
- [98] Z. Guo, C. Zheng, and B. Shi, "Discrete lattice effects on the forcing term in the lattice Boltzmann method," *Physical review E*, vol. 65, no. 4, p. 046308, 2002.

- [99] P. J. Dellar, "An interpretation and derivation of the lattice Boltzmann method using Strang splitting," *Computers & Mathematics with Applications*, vol. 65, no. 2, pp. 129-141, 2013.
- [100] F.-W. Chen and C.-W. Liu, "Estimation of the spatial rainfall distribution using inverse distance weighting (IDW) in the middle of Taiwan," *Paddy and Water Environment*, vol. 10, no. 3, pp. 209-222, 2012.
- [101] G. E. Fasshauer, *Meshfree approximation methods with MATLAB*. World Scientific, 2007.
- [102] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE transactions on acoustics, speech, and signal processing*, vol. 29, no. 6, pp. 1153-1160, 1981.
- [103] D. Khaledyan, A. Amirany, K. Jafari, M. H. Moaiyeri, A. Z. Khuzani, and N. Mashhadi, "Low-Cost Implementation of Bilinear and Bicubic Image Interpolation for Real-Time Image Super-Resolution," in *2020 IEEE Global Humanitarian Technology Conference (GHTC)*, 2020: IEEE, pp. 1-5.
- [104] G. Wissocq, J.-F. Boussuge, and P. Sagaut, "Consistent vortex initialization for the athermal lattice Boltzmann method," *Physical Review E*, vol. 101, no. 4, p. 043306, 2020.
- [105] D. Ingham and T. Tang, "A numerical investigation into the steady flow past a rotating circular cylinder at low and intermediate Reynolds numbers," *Journal of Computational Physics*, vol. 87, no. 1, pp. 91-107, 1990.
- [106] B. Fornberg, "A numerical study of steady viscous flow past a circular cylinder," *Journal of Fluid Mechanics*, vol. 98, no. 4, pp. 819-855, 1980.
- [107] M. Cheng and K. C. Hung, "Lattice Boltzmann method on nonuniform mesh," *International Journal of Computational Engineering Science*, vol. 5, no. 02, pp. 291-302, 2004.
- [108] H. Yoo, G. Wissocq, J. Jacob, J. Favier, and P. Sagaut, "Compressible lattice Boltzmann method with rotating overset grids," *Physical Review E*, vol. 107, no. 4, p. 045306, 2023.
- [109] T. Tang and D. Ingham, "On steady flow past a rotating circular cylinder at Reynolds numbers 60 and 100," *Computers & Fluids*, vol. 19, no. 2, pp. 217-230, 1991.
- [110] S. Guo, Y. Feng, and P. Sagaut, "Improved standard thermal lattice Boltzmann model with hybrid recursive regularization for compressible laminar and turbulent flows," *Physics of Fluids*, vol. 32, no. 12, 2020.